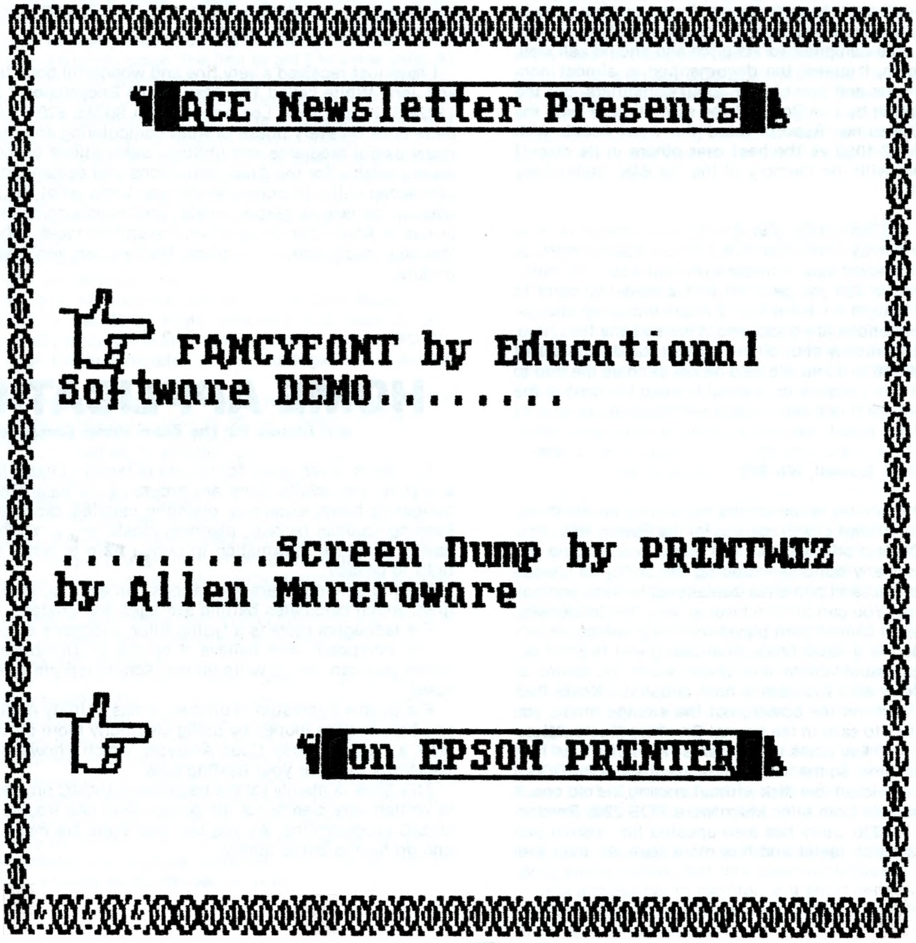
ATARI COMPUTER ENTHUSIASTS

3662 Vine Maple Dr. Eugene OR 97405

FEBRUARY, 1984

Mike Dunn & Jim Bumpas, Editors



ACE Newsletter Presents

FANCYFONT by Educational Software DEMO.......

.........Screen Dump by PRINTWIZ by Allen Marcroware

on EPSON PRINTER

News and Reviews

by Mike Dunn, Co-Editor

Another new year starts for ACE; and as time goes on, our members become more and more advanced in their programming abilities. This issue has some great programs, but little in BASIC. Much is devoted to ACTION!, the new language developed by Clint Parker and sold by O.S.S. Clint will be submitting programs to ACE; the first is a companion to a similiar program in BASIC by Jerry White in the FEB issue of ANTIC! More on that in a separate article. Stan Ockers' game this month is also in ACTION!. And Ruth Ellsworth turns to ACTION! It seems anyone who tries it really likes it, because it is easy to program with and very fast. We also have a PILOT program by Ruth and a nice machine-language utility by Greg Menke. We also have a large number of excellent articles taken from other user group newsletters which are really outstanding and we will try to include them as we have room.

The most intensive reviewing this month has been with the new Atari 800XL 64K computer. Kirt Stockwell, Jim Bumpas and I have used it alot. It has a very nice keyboard with the keys where they are supposed to be — no hitting the Atari Key by mistake. The keyboard feedback is through the TV speaker, so you can set the loudness by the volume control as you like it. The picture quality is better than the old 800 — more vivid colors with no interference, possibly because the entire power supply is now separate. The sound is also better. The BASIC is built in and has several new graphic modes (16 in all). To disable BASIC, you hold the OPTION key down when booting up. There is a built in self-test program, and a HELP key available. It also has the new expansion port, but no information seems to be available on the pin-out of the port. The computer is nice and small, and looks beautiful. It has the same OS as the 1200XL, which means there are new features but many of the older 3rd party software programs will not run without a translator disk available from Atari which converts the OS to the old kind, so that most software will run. This Translator disk is not copyrighted, so it is easy to get. It does not work with cartridges; at least I couldn't get it to work. New software coming out will work; the problem is the use of illegal entry points mostly in protection schemes. A very nice computer for its price and should sell well. To keep the price down, I guess, the documention is almost non-existent — if you get one and don't know about computers, get the book **Your Atari Computer** by Lon Poole at the same time. **Flash: the new Feb issue of Consumer Reports rated the Atari 600XL with AtariWriter and the Atari 1050 as the best over others in its class!!** They go on to say that with the memory of the the 64K model, they would like it even better.

In the Jan issue of Computers and Electronics magazine is a wonderfully written article by Randy Carlstrom for a cassette interface allowing the use of a standard tape recorder with your Atari. The basic kit costs only $20, and for $30 you get a kit with a tape-duplicator to make digital copies. I bought the kit; it took 2 hours including interuptions to make. The intructions are good, and it worked the first time. The unit does not automatically shut off, so I called Randy and asked him if it could be modified to do so. He said he will send us the info to do so, but did not put it in because he wanted to keep the cost of the unit under $20. It turns out it is a very simple modification using a $3 Radio Shack relay and a diode. He also sells assembled and tested units, as well as the bare board and parts. Write him at **RC Systems 121 West Winesap Road, Bothell, WA 98012** for more info.

The most useful program we received this month is a printer dump program by Jerry Allen, **Printwiz.** With options for the Epson, NEC, Prowriter, Gemini, and Okidata printers, this $29 program will dump any graphic screen. It has many options, including the ability to change the size of the picture, etc., and prints horizontally rather than vertically as most other dumps. You can print in inverse, vary the dot density, etc. As with all dumps, it cannot print player-missile graphics, or protected games. Included is a lister program allowing you to print out listings with the "unprintable" characters at any width you desire on the above printers. Jerry also includes a note about the **Koala Pad**: with the picture you want on the screen, not the storage mode, you can press the "＞" key to save in the normal **Graphics Master/Micropainter** format. The "＜" key loads the filename "PICTURE". All files are saved with this filename, so the names must be changed by DOS if you want more than one picture per disk without erasing the old one. A very fine program, available from Allen Macroware, POB 2205 Rendondo Beach, CA 90278 for $30. Jerry has also updated his popular program **DiskWiz II**; it is much faster and has more features than ever before.

Also received this month, but too late for a complete review, were a number of the wonderful **Tricky Tutorials** by Educational Software (formally Santa Cruz Ed. software), 4565 Cherryvale Ave., Soquel, CA 95073. These come with a disk full of programs and nice booklets of 33 to 68 pages explaining all. Included were TT#12 ($30) **The S.A.M. Tutorial**, teaching you to program the software "mouth" by Don't Ask Software. Included is a utility to add to older versions of S.A.M. to add "knobs". TT#13 **BASIC Programming Tools** contains a group of useful machine language tools allowing you to Renumber or Delete lines, Trace, Expand multi-statement lines to single statement lines for debugging, and list the program variables and constants. Also included is a lister for listing unlistable characters in listings. A lot of utility for $30. TT#14 **Advanced Programming Tools** has many sophisticated machine language sub-routes to allow BASIC programmers to add advanced manipulations to their programs such as high-speed player-missile graphic routines. It also includes the basic part of the Character Editor below for only $40. TT#15 **Fancy Font** is my favorite: it includes the Character Editor, various fonts, and tutorials and programs to make really fancy displays you won't believe! An extensive tutorial is included, as well as a utility allowing you to download any of these fonts to a Epson FX80 and print them out! Only $30, and a must if you have the FX80. You can get the set of the first six Tricky Tutorials in a 3-ring binder for $120 and #7, 8, 9, 10, 11 and 15 for $120. A wonderful way to learn to program your Atari at any level.

For a really good bargain, the Atari User group in Brooklyn, **B.A.S.I.C.**, is now publishing their Newsletter on disk for $7.50 a copy or $35 a year. Volume 1 #2 is now ready and includes a number of good programs, including an updated Jones Terminal, very good tutorials on various subjects, editorials, etc., similiar in style to MicroMedia magazine. It also includes a really fantastic pinball game made with PinBall Construction Set and in run-time format which is worth the whole year's subscription price! Write to B.A.S.I.C., c/o P. Fazio, 2724 E. 23 St, Brooklyn, NY 11235.

I have just received a very fine and wonderful book by Gary Phillips and Jerry White called **The Atari User's Encyclopedia** (The Book Co., 11223 S. Hindry Ave., Los Angeles, CA 90045, $20). This a truly complete book on every phase of Atari computering and more. It includes many useful programs and utilities, descriptions of most of the software available for the Atari, definitions and explanations of anything connected with our computer (do you know what CRC error means? what all the new XL graphic modes are? need a complete memory map or list of Atari User Groups? an Assembler move subroutine? — all this plus much more in this book). Well written, very interesting and up to date.

HOME APPLICATIONS

and Games for the Atari Home Computers

by Timothy P. Banse

This book is very good for the entire family. It has a lot of variety for everyone. For adults there are programs for balancing checkbooks, budgeting home expenses, counting calories, blood alcohol testing, keeping medical records, planning meals, and a "Crypto-System" for saving sensitive information in coded form in order to protect your right to privacy.

For the children there are various games, a number averaging program which calculates batting averages and grades.

For teenagers there is a typing tutor, a jogger's electronic logbook, music composer, and believe it or not, a "Dumb Word Processor" which you can use to write letters, school reports or the best-selling novel.

For people interested in utilities, it has a Utility Audit, which shows you how to save money by using electricity more efficiently. There is also a Heating Loss Cost Analysis which shows you heat-saving measures to lower your heating bills.

This book is mainly for the beginner in BASIC programming. I feel it is written very clearly for the person who has trouble understanding BASIC programming. As you can see, there are many things to learn and do for the entire family.

— Jackie Heden

BUMPAS REVIEWS

ATARI informs me the Synapse applications being distributed by Atari will be released March 1. These are Syn-File (database manager), Syn-Calc (spreadsheet), and Syn-Trend (including Syn-Graph and Syn-Stat). All three will be included in a package called "Syn-Apps". Atari will not be distributing Syn-Text, the new word processor, as it competes with AtariWriter. All of these applications will be compatible with AtariWriter. We'll tell you more when we see review copies.

Datasoft has announced a dozen or so new products, including a "Dallas" game and Letter Wizard, their new word processor. Strategic Simulations will release Questron for the Atari this Spring, too.

Epyx, 1043 Kiel Ct., Sunnyvale, CA 94089, has released a package of "arcade classics" (price not included). The disk contains two arcade shoot 'em ups, "Starfire" and "Fire One".

Starfire looks and plays like the ship-to-ship combat in StarRaiders. The player flies through space, lining up wave after wave of enemy tactical fighters trying to protect "Exidy space freighters" from destruction by your "Starcraft". The screen shows the enemy and various missiles coming at you. Below there is a "radar screen" showing the approach of nearby enemy. Points gained will automatically refuel you at various levels.

Fire One is a confusing battle between your submarine and an enemy sub. The screen is split (when you're on the surface) into a sonar display and a periscope display. Both displays are full of ships at start. You must sink enemy ships and the enemy sub before he sinks you and yours. You run out of torpedoes from time to time, but they automatically reload. When submerged, you have only the sonar to guide you.

SILICON WARRIOR is another program from Epyx on cartridge. It is a combination of Tic-Tac-Toe and Mr. Cool. The scenario of the game is built around warrior wizards who must "program" 5 chips in a row on a 5x5 grid.

For 1-4 players, the computer may play one or more non-player positions. Up to 7 skill levels and 3 speeds of play may also be selected. The higher levels add "black holes", "laser fire", and "shields" in different combinations.

The players jump to empty chips, turning an enemy color to neutral or a neutral color to a friendly color. The first to get 5 in a row wins. At the higher levels which use lasers and shields, a player may return to the "power pyramid" of your color to restore lost power. At the fastest speed this game will challenge the most agile arcade player. At any speed it will challenge a player to develop good strategy.

Drol (Broderbund, 17 Paul Dr., San Rafael, CA 94903) is an arcade game where you shoot scorpions, monsters, snakes, daggers, swords, arrows, balloons, helicopters, witch doctors, and turkeys before you can rescue (by bumping your character into them) girls, boys, lizards, crocodiles, and mothers.

The playfield scrolls across approximately 6 screens and the character graphics are very well drawn.

SOLO FLIGHT ($32, Microprose, 10616 Beaver Dam Road, Hunt Valley, MD 21030) is the **BEST** flight simulator I've seen for any microcomputer. And I include the Microsoft simulator for the IBM PC. I have not yet seen Flight Simulator II from Sublogic, but it will be tough to top this one.

The top half of the screen shows the view out of the cockpit. The forward view shows a heads up display of your plane on the windshield as though you were flying behind yourself and can see the attitude your plane is in. The display also puts a shadow beneath the plane when close to the ground. The views to right, left and rear are unobstructed (bubble cockpit, I guess).

The lower half of the screen shows the instrument display, including throttle (10 settings), landing gear and brake lights, flaps (3 settings), altimeter, airspeed, artificial horizon, pitch, digital and alpha directional compass, rate of climb, fuel gauge, temperature warning light, two VOR (for instrument flying) indicators, and a glide path indicator. The instruments also show ceiling and direction and force of wind. You can see you have pretty complete instrumentation. They are easy to read and are not crowded.

You may fly on any one of three maps: Kansas (flat, easiest for beginners); Oregon/Washington (the Cascades offer some elevation problems); and Colorado (those little valley airports are a challenge for the expert). You may select practice flying, or play the mail pilot game where you must deliver mail bags to up to six other cities. You may select one of four levels of pilot skill from Student to Command Pilot. Choice of weather includes Clear, Windy, or IFR (for instrument flying). Landing practice places the aircraft on short final approach.

The program may be paused at any time, and emergency equipment failures may be created for practice purposes. You can fly into the clouds by gaining altitude above the ceiling level.

At the beginning of a trip the screen displays the map of the state, with 7 airfields, topographical indications, cities and VOR stations. When the trip is over the map reappears with your route (sometimes quite convoluted if you've had trouble lining up a final approach) drawn out on the screen. Your speed is indicated on the map, too. If the dots are spaced close together, your speed was relatively slow. If they are spaced farther apart, your speed was better. The program scores your performance with a point system.

I think there are some skills here which one might want to master before getting into a plane to learn to fly. I've already destroyed several hundred thousand dollars worth of aircraft (and no telling how many lives!) trying to land. But it's a real feeling of accomplishment when you successfully land and deliver the mail. If you enjoy aircraft simulators, I don't believe you can find a better one than this for your computer.

The **Astra 1620** disk drives are now being delivered. I tried two of them myself. The first one only worked for about a half an hour before it refused to read any disk. I turned it off for awhile, and then I found I could read disks again for another half hour before it stopped again. Disks removed from the drive were noticeably warm. I think the heat caused the problem.

The second one I tried worked fine for the two days I had it. But disks removed from this one seemed to me just as warm as before. I am used to my disks seeming as cool when I remove them from a drive as when I inserted them. The warmed-up disks alarm me too much to be comfortable with that drive. So I returned it, too.

The Astra folks are very nice. They are responsive to users with questions or who may need help. They chose to put the power supply into the disk drive case. Most disk drives for the Atari seem to have external power supplies. But there are many disk drives with internal power supplies. The heat build up in the Astra might be improved with a better design for dissipating the heat. Mounting the drives vertically, in a case with more ventilation, might help.

The $595 price for 2 double-density disk drives is excellent. DOS XL is included, with excellent documentation. The Indus drive also provides DOS XL, but the documentation Insus provides is not adequate, in my view, to make full use of that DOS. And the Rana and Trak drives don't come with any DOS at all, although Trak does provide a program which gives one software control over density. Trak and Indus provide dip switches giving a user hardware control of default densities. I've heard the Astra being called a "toaster" already. And the question about heat in this drive is the ONLY thing which makes me hesitate to recommend this drive.

—Jim Bumpas

BASIC XL

A review by Graham Smith

There are several versions of Basic for the Atari now on the market, and some of them offer improvements at the expense of not always being compatible with existing Atari Basic software. Basic XL is an exception. Since it uses the same tokens as Atari Basic most of my programs — with the exception of Basic Commander — ran without difficulty

Basic XL comes on a 24k cartridge. But because of a neat bit of "bank switching" it does not use up any more memory than Atari Basic. It is, however, a **MUCH** more extended basic language. Furthermore, it is easier to use and much more complete than Atari Basic.

Special features include auto-numbering, renumbering, mass-delete, trace, Lvar (List Variables), as well as the ability to access the disk menu and manipulate programs without losing the program in memory. I find I seldom use the DOS menu anymore. Needless to say, Basic XL handles strings with ease and most variables are automatically dimensioned to 40 characters unless the user specifies otherwise.

One of the features of this program is a "fast" command which precompiles the "GOSUBs" and "GOTOs" by address rather than line numbers. This considerably speeds up programs with many subroutines. My own comparison test shows Basic XL usually runs faster than Atari Basic, however, this varies with the type of task and length of program.

The documentation in this package is superb. The first section is a 176 page Basic tutorial (one of the authors is Bill Wilkinson). The second part of the manual is a straight forward description of all the commands — old and new. It is all very thorough and and well presented.

I am enthusiastic about this package. I never cared much for Basic programing before, but this powerful language has won me over.

It is impossible to mention in a short review all the advantages, but in addition to improving and simplifying Atari Basic there are no less than 46 new commands.

This is not just another Basic. I wouldn't be without it.

— Graham Smith
Ace Eugene

GETTING INTO ACTION!

(A "Non-expert" Looks at the ACTION! language)
by Ruth Ellsworth

Every programmer dreams of finding a computer language which is fast, powerful, and user-friendly. In my opinion, ACTION! comes as close to filling the bill as any programming language I have tried.

ACTION! is attractively packaged. From the yellow looseleaf binder to the orange cartridge, the fact that everything connected with this system is well done is obvious. The documentation is excellent, but beginners should probably wait for some tutorials and at least the Programmer's Disk which is now available. For the intrepid novice who has some experience in BASIC or LOGO, a careful reading and working through the examples will allow one to begin using ACTION! without the Programmer's Disk which I am anxiously awaiting.

While the documentation is not indexed, it is divided into parts which are then divided into topics. There are five main parts, not counting the introduction and appendices: the Editor, the Monitor, the Language, the Compiler, and the Library. If one has some idea as to which part pertains to his problem, it is not difficult to find what one is looking for.

The ACTION! Editor is without a doubt the most powerful and easiest to use editor I have found. It contains all the commands which make the ATARI editor so superior plus a number of additional commands which allow the user to move easily through programming text. The unique commands are not only easy to use, but easy to remember. For example, control-shift-R allows one to Read a file from a peripheral device into the Editor.

Without an autorun system, ACTION! boots directly into the Editor. The Monitor is accessed by command from the Editor. Programs can be run directly from a peripherial device to the Monitor, or compiled from the Editor then run. Programs are run and immediate mode commands are executed in the Monitor.

The ACTION! Language section of the documentation is especially well done. It is loaded with examples and explanations, making it the finest language manual I have seen. The wording is clear and surprisingly nontechnical. Some of the explanations of basics, such as Loops and Modules, are more easily understood than similar explanations I have read in books written for beginners.

The Compiler is explained, and the Library functions allowing the use of many routines the user does not have to define are contained in their respective parts of the documentation. The Library Part of the manual is its only weakness. A few examples would have helped greatly. One hopes the Programmer's Disk will make up for this lack. However, even without examples, it is possible to use the library routines if one persists.

I have chosen to use a joystick sketch program for a sample comparison of ACTION!, BASIC, PILOT, and LOGO. FORTH is not included because my documentation is presently loaned and I am somewhat rusty in its use. I will take the liberty of making some comment as to the difference between ACTION! and FORTH without the specifics of a program example. I realize, also, it is possible to make a joystick etch in PILOT with many fewer lines a la Jim Carr's routine. I have tried to make these programs as similar as possible in logic for the purposes of comparison rather than program length.

The first impression one has when using ACTION! is speed. Compared to the other languages available for the ATARI, ACTION! is faster than FORTH, much faster than BASIC or PILOT (even in its speedy graphics mode), and very much faster than LOGO.

ACTION! is basically a version of PASCAL. It is modular, built in pieces called Procedures put together to make a program. The major advantage I see in it over FORTH is the syntax is much easier to use and understand (I was not able to test the greater speed of ACTION!). FORTH does have an advantage of being transportable to other machines which should not be overlooked, however. Because it is modular, and the form in which procedures are written are similar, one acquainted with LOGO will find ACTION! surprisingly friendly. PILOT and BASIC programmers may find structure strange, a book liket- **PASCAL FOR BASIC PROGRAMMERS**tby Seiter and Weiss might be helpful to those intimidated by the differences. The ACTION! manual probably will be sufficient for more experienced BASIC programmers.

The programs which follow are all basically the same. Except for LOGO, the location of the X and Y co-ordinates are given as variables at the beginning of the program (ATARI LOGO seems to have some reservations about accepting :X :Y in any form as SETPOS or POS inputs, I have used the setheading command, trying to keep the program structure as similar as possible). The graphics mode is set either before or after the variables, if necessary. Turtle graphics is its own mode and is, therefore, not called in LOGO or PILOT. The program loop comes next. It becomes infinite in BASIC through the use of GOTO, in PILOT by jumping back to the main module (*STICK), in LOGO throug the procedure calling itself (JOYDRAW), and in ACTION! by the use of the empty DO — OD loop.

ACTION! is so fast that one will note the difference in the delay loop between BASIC and ACTION!. One will also note the similarity, and yet subtle differences in the commands between ACTION! and BASIC (color, graphics, and stick commands for example).

The thing about writing graphics routines in ACTION! which seem a little unusual to me is that the X and Y co-ordinates had to appear as variables. One does not Plot(number,number) in ACTION! procedures as far as I can tell. Other than that small problem which was easily overcome, I found ACTION! to be a programmer's dream. I do not expect to reach the expertise of Stan Ockers et al, but I do expect to do many things with ACTION! It is one programming language which, in my opinion, will probably be preferred by ATARI users everywhere.

TIDBITS

80 Columns for Tinytext 2.0

This article is intended for those of you with Video 80 from the April 1983 Compute!. For those not aware of this incredible program, it gives one an 80 column display through software. To use it, it must first be loaded from DOS using option L with Atari DOS (LOA from OS/A+). Then you can run a modified version of Tinytext 2.0 and the display option will show you your text in 80 columns. Listing 1 contains the changes to make to Tinytext in order to make it work with Video-80. In addition to adding those lines, you should delete lines 590, 726, 728, 840, and 890.

Some comments on using Video-80 with Tinytext: First, since Video-80 uses a graphics 8 display, it eats a lot of memory we could use to store text. In fact, don't even think of using it with Tinytext unless you have at least 40K memory. As well, the characters Video-80 produces are hard to read on many monitors, but I think the advantage of being able to see what your page will look like more than makes up for this inconvenience.

If you have OS/A+ 2.10, you could set up a STARTUP.EXC file which will run Video 80 and then Tinytext automatically. To do this we will need the following: an AUTORUN.SYS file which runs a BASIC program automatically, the Video-80 file (with the name VIDEO80.OBJ), and a STARTUP.EXC file which will instruct OS/A+ what to do when we power up. Listing 2 contains a program to make such a STARTUP.EXC file, and Listing 3 will produce an AUTORUN.SYS file to automatically load and run a BASIC program on disk with the name 'HELLO'. Okay, to start off, initialize and write DOS to a blank disk. Now type in Listing 2 and RUN it. Now we need to think about what we want OS/A+ to do when we power up. Well, we want it to LOA VIDEO80.OBJ, and then run Tinytext. To get it to run Tinytext, we will tell it to LOA AUTORUN.SYS and then go to the CARtridge. Therefore, our STARTUP.EXC program looks like this:

```
LOA VIDEO80.OBJ
LOA AUTORUN.SYS
CAR
```

Use Listing 2 to create such a STARTUP.EXC file.

Now we need an AUTORUN.SYS file. You may use Listing 3 to create one if you don't already have one in your library. Now put your Video 80 file on the disk. Finally, load your copy of Tinytext 2.0 and make the corrections, and then save it on the disk with whatever name your AUTORUN.SYS file wants to run (if you used listing 3, save it as "D:HELLO"). On your disk you should have the following files: DOS.SYS, AUTORUN.SYS, STARTUP.EXC, VIDEO80.OBJ, and Tinytext (SAVEd as HELLO or whatever).

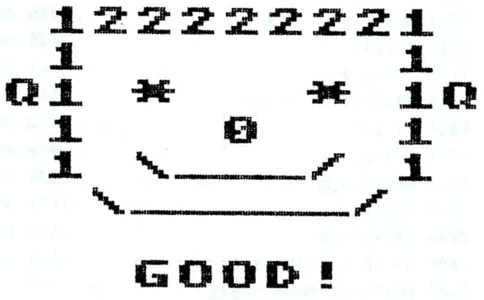
You should now have a working copy of what I call Super Tinytext. Reboot and enjoy your 80 columns.

The principle used in the modification of Tinytext deserves some discussion. By changing the device we OPEN the file to, we can completely change the operation of the program. I have used this several times when debugging programs which send output to the printer via a PRINT #1. While debugging, I simply change the line near the beginning of the program from OPEN #1,8,0,"P"to OPEN #1,8,0,"E"or OPEN #1,8,0,"V". In this way I can see what the output will look like without wasting reams of paper. Then, when everything looks OK, I just change the "E" or "V" back to a "P", thereby redirecting the output to the printer. In this modified Tinytext, we do exactly the same thing. If the user wants to display the text, we open the IOCB to Video 80. If printing is chosen, we open it to the printer. After opening it, exactly the same routines are executed to print out the text.

— Dale Lutz
CANADA

Listing in BASIC and Pilot, as well as
Screen Dumps by PRINTWIZ by Allen
Macroware

Ruth Ellsworth: JOYSTICKS COUNT



```
10 R:JOYSTICK COUNT
20 R:BY RUTH ELLSWORTH
30 R:ACE NEWSLETTER 3662 VINEMAPLE
35 R: EUGENE, OR 97405   $12 YEAR
40 R:FEB 1984
50 J:*BEGIN
60 *BOAT
70 GR:PENRED;TURNTO90;DRAW10;TURNTO45;
DRAW4;TURNTO270;DRAW16;TURNTO135;DRAW4
;TURNTO315;FILL4
80 GR:TURNTO90;GO6
90 C:@B709=174+0
100 C:@B710=0+0
110 GR:PENYELLOW;TURNTO0;DRAW11;TURNTO
135;DRAW11;TURNTO270;DRAW7;TURNTO0;FIL
L6
120 E:
130 *HOWMANY
140 VNEW:
150 GR:QUIT
160 C:@B1373=0
170 C:@B1374=2
180 WRITE:5,
190 WRITE:5,
200 WRITE:5,
210 WRITE:5, how many boats?
220 PA:100
230 CLOSE:5
240 GR:CLEAR
250 C:@B710=0+0
260 *START
270 C:@B764=255
280 C:#A=?\5
290 J(#A=0):*BOAT1
300 J(#A=1):*BOAT2
310 J(#A=2):*BOAT3
320 J(#A=3):*BOAT4
330 J(#A=3):*BOAT4
340 J(#A=4):*BOAT5
350 E:
360 *NUM1
370 GR:TURNTO180;DRAW6
380 GR:TURN90;DRAW1
390 GR:TURN90;DRAW5
400 GR:TURNTO180;GO1
410 GR:TURN90;DRAW1
420 GR:TURNTO180;GO4
430 GR:TURN90;DRAW1
440 GR:TURN180;DRAW6
450 E:
460 *NUM2
470 GR:TURNTO45;DRAW1
480 GR:TURN45;DRAW3
490 GR:TURN90;DRAW1
500 GR:TURN45;DRAW5
510 GR:TURNTO90;DRAW1
520 GR:TURN-45;DRAW5
530 GR:TURNTO180;GO4
540 GR:TURNTO270;DRAW5
550 E:
560 *NUM3
570 GR:TURNTO90;DRAW6
580 GR:TURN135;DRAW3
590 GR:TURN45;DRAW1
600 GR:TURNTO45;DRAW2
610 GR:TURNTO180;DRAW2
620 GR:TURNTO90;DRAW1
630 GR:TURNTO180;DRAW2
640 GR:TURNTO45;DRAW1
650 GR:TURN180;DRAW1
660 GR:TURNTO270;DRAW4
670 GR:TURNTO0;DRAW1
680 GR:TURNTO-90;DRAW1
690 E:
700 *NUM4
710 GR:TURNTO180;DRAW6
720 GR:TURN90;DRAW1
730 GR:TURN90;DRAW6
740 GR:TURNTO-135;DRAW5
750 GR:TURNTO180;DRAW1
760 GR:TURN-90;DRAW1
770 GR:TURNTO0;DRAW1
780 GR:TURNTO45;DRAW3
790 GR:TURNTO225;GO5
800 GR:TURNTO90;DRAW6
810 E:
820 *NUM5
830 GR:TURNTO90;DRAW5
840 GR:TURN180;GO5
850 GR:TURN-90;DRAW3
860 GR:TURNTO90;DRAW1
870 GR:TURNTO0;DRAW2
880 GR:TURNTO180;GO2
890 GR:TURNTO90;DRAW3
900 GR:TURNTO180;DRAW3
910 GR:TURNTO0;GO1
920 GR:TURNTO90;DRAW1
930 GR:TURNTO0;DRAW1
940 GR:TURN180;GO1
950 GR:TURNTO270;GO5
960 GR:TURN180;DRAW1
970 GR:TURNTO180;DRAW1
980 GR:TURNTO90;DRAW3
990 E:
1000 *BOAT1
1010 GR:GOTO-50,30
1020 U:*BOAT
1030 J:*MATCH1
1040 E:
1050 *BOAT2
1060 GR:GOTO-50,30
1070 U:*BOAT
1080 GR:GOTO30,30
1090 U:*BOAT
1100 J:*MATCH2
1110 E:
1120 *BOAT3
1130 GR:GOTO-50,30
1140 U:*BOAT
1150 GR:GOTO-7,30
1160 U:*BOAT
1170 GR:GOTO30,30
1180 U:*BOAT
```

```
1190 J:*MATCH3
1200 E:
1210 *BOAT4
1220 GR:GOTO-50,30
1230 U:*BOAT
1240 GR:GOTO-7,30
1250 U:*BOAT
1260 GR:GOTO30,30
1270 U:*BOAT
1280 GR:GOTO-32,7
1290 U:*BOAT
1300 J:*MATCH4
1310 E:
1320 *BOAT5
1330 GR:GOTO-50,30
1340 U:*BOAT
1350 GR:GOTO-7,30
1360 U:*BOAT
1370 GR:GOTO30,30
1380 U:*BOAT
1390 GR:GOTO-32,7
1400 U:*BOAT
1410 GR:GOTO9,7
1420 U:*BOAT
1430 J:*MATCH5
1440 E:
1450 *MATCH1
1460 PA:30
1470 J(@B764=31):*RIGHT
1480 J(@B764<>31):*MATCH1
1490 E:
1500 *MATCH2
1510 PA:30
1520 J(@B764=30):*RIGHT
1530 J(@B764<>30):*MATCH2
1540 E:
1550 *MATCH3
1560 PA:30
1570 J(@B764=26):*RIGHT
1580 J(@B764<>26):*MATCH3
1590 E:
1600 *MATCH4
1610 PA:30
1620 J(@B764=24):*RIGHT
1630 J(@B764<>24):*MATCH4
1640 E:
1650 *MATCH5
1660 PA:30
1670 J(@B764=29):*RIGHT
1680 J(@B764<>29):*MATCH5
1690 E:
1700 *RIGHT
1710 U(#A=0):*RONE
1720 U(#A=1):*RTWO
1730 U(#A=2):*RTHREE
1740 U(#A=3):*RFOUR
1750 U(#A=4):*RFIVE
1760 PA:100
```

```
2350 PA:100
2360 POS:2,3
2370 WRITE:5,2  * *
2380 PA:100
2390 POS:2,5
2400 WRITE:5,3  * * *
2410 PA:100
2420 POS:2,7
2430 WRITE:5,4  * * * *
2440 PA:100
2450 POS:2,9
2460 WRITE:5,5  * * * * *
2470 PA:200
2480 WRITE:5,↰
2490 WRITE:5, YOUR TURN TO
2500 WRITE:5, MAKE STARS
2510 WRITE:5, PUSH THE TRIGGER
2520 PA:100
2530 WRITE:5,↰
2540 C:#C=0
2550 *TRIGGER
2560 J(%T0=0):*TRIGGER
2570 J(%T0=1):*COUNT
2580 E:
2590 *COUNT
2600 C:#C=#C+1
2610 J(#C=1):*ONE
2620 J(#C=2):*TWO
2630 J(#C=3):*THREE
2640 J(#C=4):*FOUR
2650 J(#C=5):*FIVE
2660 E:
2670 *ONE
2680 POS:5,3
2690 WRITE:5,1
2700 POS:5,5
2710 WRITE:5,*
2720 PA:20
2730 J:*TRIGGER
2740 E:
2750 *TWO
2760 POS:5,3
2770 WRITE:5, _
2780 POS:7,3
2790 WRITE:5,2
2800 POS:7,5
2810 WRITE:5,*
2820 PA:20
2830 J:*TRIGGER
2840 E:
2850 *THREE
2860 POS:7,3
2870 WRITE:5, _
2880 POS:9,3
2890 WRITE:5,3
2900 POS:9,5
2910 WRITE:5,*
2920 PA:20
```

```
1770 VNEW:
1780 GR:QUIT
1790 C:@B1373=0
1800 C:@B1374=2
1810 WRITE:5,
1820 WRITE:5,      122222221
1830 WRITE:5,      1       1
1840 WRITE:5,     Q1 *   * 1Q
1850 WRITE:5,      1   0   1
1860 WRITE:5,      1 \___/ 1
1870 WRITE:5,        \_____/
1880 WRITE:5,
1890 WRITE:5,        good!
1900 PA:100
1910 WRITE:5,↰
1920 J:*BOATMOD
1930 E:
1940 *RONE
1950 GR:GOTO-7,7
1960 U:*NUM1
1970 E:
1980 *RTWO
1990 GR:GOTO-7,7
2000 U:*NUM2
2010 E:
2020 *RTHREE
2030 GR:GOTO-7,7
2040 U:*NUM3
2050 E:
2060 *RFOUR
2070 GR:GOTO-7,-16
2080 U:*NUM4
2090 E:
2100 *RFIVE
2110 GR:GOTO-7,-16
2120 U:*NUM5
2130 E:
2140 *BEGIN
2150 C:@B1373=0
2160 C:@B1374=2
2170 *TITLE
2180 WRITE:5,
2190 WRITE:5,
2200 WRITE:5,
2210 WRITE:5,
2220 WRITE:5, **joystick count**
2230 PA:200
2240 C:$NAME=DAN
2250 WRITE:5,↰
2260 POS:2,2
2270 WRITE:5,  HELLO, $NAME,
2280 POS:1,5
2290 WRITE:5,LET'S COUNT STARS
2300 WRITE:5,
2310 PA:200
2320 WRITE:5,↰
2330 POS:2,1
2340 WRITE:5,1  *
```

```
2930 J:*TRIGGER
2940 E:
2950 *FOUR
2960 POS:9,3
2970 WRITE:5, _
2980 POS:11,3
2990 WRITE:5,4
3000 POS:11,5
3010 WRITE:5,*
3020 PA:20
3030 J:*TRIGGER
3040 E:
3050 *FIVE
3060 POS:11,3
3070 WRITE:5, _
3080 POS:13,3
3090 WRITE:5,5
3100 POS:13,5
3110 WRITE:5,*
3120 PA:20
3130 J:*BOATMOD
3140 E:
3150 *BOATMOD
3160 WRITE:5,K
3170 WRITE:5, NOW, DAN,
3180 WRITE:5, LET'S COUNT FIVE
3190 WRITE:5,
3200 WRITE:5,        BOATS
3210 WRITE:5,
3220 WRITE:5, PUSH THE TRIGGER
3230 PA:60
3240 CLOSE:5
3250 GR:CLEAR
3260 C:#C=0
3270 *BTRIGGER
3280 J(%T8=0):*BTRIGGER
3290 J(%T8=1):*BCOUNT
3300 E:
3310 *BCOUNT
3320 C:#C=#C+1
3330 U(#C=1):*BONE
3340 U(#C=2):*BTWO
3350 U(#C=3):*BTHREE
3360 U(#C=4):*BFOUR
3370 U(#C=5):*BFIVE
3380 PA:20
3390 J:*BTRIGGER
3400 E:
3410 *BONE
3420 GR:GOTO-50,30
3430 U:*BOAT
3440 GR:GOTO-47,25
3450 U:*NUM1
3460 E:
3470 *BTWO
3480 GR:GOTO30,30
3490 U:*BOAT
3500 GR:GOTO31,25
3510 U:*NUM2
3520 E:
3530 *BTHREE
3540 GR:GOTO-7,7
3550 U:*BOAT
3560 GR:GOTO-7,2
3570 U:*NUM3
3580 E:
3590 *BFOUR
3600 GR:GOTO-50,-20
3610 U:*BOAT
3620 GR:GOTO-47,-25
3630 U:*NUM4
3640 E:
3650 *BFIVE
3660 GR:GOTO30,-20
3670 U:*BOAT
3680 GR:GOTO30,-25
3690 U:*NUM5
3700 PA:100
3710 J:*HOWMANY
3720 E:
```

Greg Menke: FUNCTION KEY EDITOR

```
1 REM ATARI COMPUTER ENTHUSIASTS
2 REM      3662 VINE MAPLE DR
3 REM EUGENE, OR 97405  $12 YEAR
4 REM           FEB 84
10 GRAPHICS 0:SETCOLOR 2,0,0
11 ? ,"FUNCTION KEY EDITOR":? ,"  by G
reg Menke.":? ,"    12/12/83"
12 ? :?
20 DIM A$(16),B$(16),C$(16),F$(20),X$(
10),D$(14)
30 ? :? "Do you want a ";CHR$(193);"ut
orun file or a":? "normal ";CHR$(194);
"inary load file ";:INPUT X$
40 IF LEN(A$)>1 THEN 30
50 IF X$="A" THEN F$="D:AUTORUN.SYS"
60 IF X$="B" THEN ? :? "Enter filename
......":INPUT F$
70 ? :? "Do you want the default messa
ges.     (Y/N) ";:INPUT X$
80 TRAP 90:XIO 36,#1,0,0,F$
90 OPEN #1,8,0,F$:RESTORE 1000
100 IF X$="Y" THEN 500
105 A$(1,1)=CHR$(156):B$(1,1)=CHR$(156
):C$(1,1)=CHR$(156):TRAP 40000
110 ? :? "Enter START message....":INP
UT D$:A$(2)=D$
120 ? :? "Enter SELECT message....":IN
PUT D$:B$(2)=D$
130 ? :? "Enter OPTION message....":IN
PUT D$:C$(2)=D$
140 FOR G=LEN(A$)+1 TO 16:A$(G,G)="♥":
NEXT G:FOR G=LEN(B$)+1 TO 16:B$(G,G)="
♥":NEXT G
150 FOR G=LEN(C$)+1 TO 16:C$(G,G)="♥":
NEXT G
160 READ A:IF A=-2 THEN 180
170 PUT #1,A:GOTO 160
180 FOR G=1 TO LEN(A$):PUT #1,ASC(A$(G
,G)):NEXT G:FOR G=1 TO LEN(B$):PUT #1,
ASC(B$(G,G)):NEXT G
190 FOR G=1 TO LEN(C$):PUT #1,ASC(C$(G
,G)):NEXT G
200 GOTO 900
491 REM
492 REM
493 REM
500 READ A:IF A=-1 THEN 900
510 IF A=-2 THEN 500
520 PUT #1,A:GOTO 500
896 REM
897 REM
898 REM
900 ? CHR$(125);"File complete...."
905 PUT #1,224:PUT #1,2:PUT #1,225:PUT
 #1,2:PUT #1,0:PUT #1,6:CLOSE #1
910 XIO 35,#1,0,0,F$
920 ? :? "Press :":? "          SELECT t
o reboot and try it!"
930 ? "          START to end."
940 IF PEEK(53279)=5 THEN POKE 53279,7
:A=USR(58487)
950 IF PEEK(53279)=6 THEN ? :CLR :END
960 GOTO 940
1000 DATA 255,255,0,6,216,6,165,12,141
,167,6,165,13,141,168
1010 DATA 6,169,22,133,12,169,6,133,13
,32,28,6,96,32,28
1020 DATA 6,108,167,6,162,6,160,45,169
,6,32,92,228,169,0
1030 DATA 133,203,96,76,95,228,165,203
,201,1,240,29,173,31,208
1040 DATA 201,7,240,22,174,251,2,224,1
,208,15,201,6,240,25
1050 DATA 201,5,240,24,201,3,240,23,76
,42,6,173,31,208,201
1060 DATA 7,208,211,169,0,133,203,76,4
2,6,76,103,6,76,114
1070 DATA 6,76,125,6,169,6,133,207,169
,169,133,206,76,133,6
1080 DATA 169,6,133,207,169,185,133,20
6,76,133,6,169,6,133,207
1090 DATA 169,201,133,206,169,1,133,20
3,160,0,177,206,201,0,240
1100 DATA 19,32,164,246,165,206,201,25
5,240,5,230,206,76,137,6
1110 DATA 230,207,76,154,6,76,42,6,0,0
,-2,156,76,79,65,68
1120 DATA 32,34,68,0,0,0,0,0,0,0,0,156
,83,65,86
1130 DATA 69,32,34,68,0,0,0,0,0,0,0,0,
156,82,85
1140 DATA 78,32,34,68,0,0,0,0,0,0,0,0,
0,-1
```

Dale Lutz: A.B.I.S.

```
1 REM  ACE NEWSLETTER
2 REM 3662 VINE MAPLE DR
3 REM EUGENE, OR 97405
4 REM  FEB 84  $12 YR
5 REM ****************************
     *                          *
     *         A.B.I.S.         *
6 REM *                          *
     *      by Dale Lutz        *
     *                          *
7 REM ****************************
10 GOSUB 100
20 IF PEEK(764)<>255 THEN GET #1,A:IF
A<>125 THEN PUT #2,A
25 IF PEEK(53279)=0 OR PEEK(53279)=1 T
HEN 300
30 GOTO 20
100 GRAPHICS 0:OPEN #1,4,0,"K":INC=5:L
INE=100
120 DIM FI$(19),T$(40)
125 ? CHR$(125):? :? " Save to Disk or
 Cassette?";:GET #1,A:? CHR$(A+128):IF
 A<>68 AND A<>67 THEN 125
126 REM D and C in Disk and Cassette i
n Inverse
130 IF A=68 THEN ? :? " What name do y
ou want it saved by":INPUT T$:FI$(3)=T
$:FI$(1,2)="D:":GOTO 140
135 ? :? " Please position the cassett
e,":? "press PLAY/REC and then RETURN"
:FI$="C:"
136 REM PLAY/REC and RETURN in Inverse
140 OPEN #3,8,0,FI$:OPEN #2,12,0,"E"
145 ? #3;LINE;" GRAPHICS 0:POKE 752,1:
C6=6:POKE 82,2:POKE 83,39:GOTO ";LINE+
INC*8:LINE=LINE+INC:WAIT=LINE
150 ? #3;LINE;" POKE 764,255":LINE=LIN
E+INC
155 ? #3;LINE;" IF PEEK(764)<>255 THEN
";LINE+INC*4:LINE=LINE+INC
160 ? #3;LINE;"POS. YP,XP:A=A+1:IF A<9
 THEN ?";CHR$(34);CHR$(160);CHR$(34);"
;":LINE=LINE+INC
165 ? #3;LINE;"IF A>8 THEN ?";CHR$(34)
;" ";CHR$(34);";:IF A>18 THEN A=0":LIN
E=LINE+INC
170 ? #3;LINE;"GOTO ";WAIT+INC:LINE=LI
NE+INC
175 ? #3;LINE;" A= PEEK(764):IF A=6 OR
 A=134 OR A=52 THEN GOTO LAST":LINE=LI
NE+INC
180 ? #3;LINE;"GOTO NEX":LINE=LINE+INC
:THIS=LINE
185 POKE 53774,112:POKE 16,112
190 RETURN
199 REM ROUTINE TO WRITE A PROGRAM TO
SAVE PRESENT SCREEN AND DO A NEW ONE
200 XP=PEEK(84):YP=PEEK(85):POKE 559,0
:POKE 752,1:? :POKE 752,0
205 FR=FR+1:? #3;LINE;" REM *** FRAME
";FR;" ***":LINE=LINE+INC:LAST=THIS:TH
IS=LINE:POSITION 2,0
210 ? #3;LINE;" ? CHR$(125);":FOR Y=1
TO 23:LINE=LINE+INC
220 ? #3;LINE;
230 P=0:FOR X=2 TO 39:GET #6,A:T$(X-1)
=CHR$(A):IF A<>32 THEN P=1
232 NEXT X
235 IF P THEN ? #3;"?#C6;";CHR$(34);T$
;CHR$(34);";":GOTO 240
237 ? #3;"?#C6";:IF Y=24 THEN ? #3;";"
;
238 ? #3
240 NEXT Y
250 LINE=LINE+INC:NEX=LINE+INC
260 ? #3;LINE;" NEX=";NEX;":LAST=";LAS
T;":XP=";XP;":YP=";YP;":GOTO ";WAIT:LI
NE=LINE+INC
262 ? #3;LINE;"?CHR$(125):POKE 764,255
:POKE 752,0:REM THIS IS THE LAST LINE"
265 POKE 764,255:POKE 559,34:POKE 712,
12
270 IF PEEK(764)=255 THEN 270
275 POKE 764,255:POKE 712,0
280 POSITION 2,2:? CHR$(125);:RETURN
300 IF PEEK(53279)<>0 THEN GOSUB 200:G
OTO 20
305 GOSUB 200:? :? "ARE YOU SURE YOU W
ANT TO QUIT THIS ":? "SESSION (Y/N)";:
GET #1,A
310 IF  NOT (A=89 OR A=121 OR A=249 OR
 A=217) THEN ? CHR$(125);:GOTO 20
320 GRAPHICS 0:POSITION 2,6:? "ENTER "
;CHR$(34);FI$;CHR$(34):? :? :? :? "POK
E 842,12:?CHR$(125)"
330 POSITION 2,0:POKE 842,13:NEW
```

Dale Lutz: TIDBITS 3

```
0 REM LISTING 1 for Dale Lutz TidBits3
14 DIM S$(45),I$(120),A$(128):SIZ=FRE(
0)-7800:DIM T$(SIZ):FOR I=1 TO 45:READ
 A:S$(I)=CHR$(A):NEXT I
115 MODE=1:LINT=1:PG=0:DIM OP$(1)
534 IF OP=3 THEN ? "PRINT":OP$="P"
538 IF OP=5 THEN ? "DISPLAY":OP$="V"
690 OPEN #1,8,0,OP$
705 OP=3
710 LINE=0:FL=0
725 IF B=19 AND LINE<=(PS-FF) THEN ? #
1;" ":LINE=LINE+1
727 IF B=16 THEN FOR I=1 TO PS-LINE:?
#1;" ":NEXT I:LINE=0:IF PG THEN GOSUB
880
830 IF OP=3 THEN ? #1;SP$(1,SP);A$:IF
LINT-1 THEN FOR I=1 TO LINT-1:? #1:NEX
T I
855 IF FL AND OP$="V" THEN GOSUB 980:C
LOSE #1:GRAPHICS 0:OP=5:GOTO 500
857 IF FL THEN CLOSE #1:GOTO 500
870 FOR I=1 TO FF:IF OP=3 THEN ? #1
875 NEXT I:IF  NOT (PG) THEN LINE=1:RE
TURN
880 PAGE=PAGE+1:SP=LL-(LEN(I$)+INT(CLO
G(PAGE))+6)+LM
885 ? #1;SP$(1,SP);I$;"Page ";PAGE:? #
1:? #1:GOTO 895
960 CLOSE #3:POKE 82,0:POKE 752,0:CLOS
E #4:GOTO 500

0 REM LISTING 2 for Dale Lutz Tidbits
5 REM PROGRAM TO MAKE A STARTUP.EXC
6 REM FILE
10 OPEN #1,8,0,"D:STARTUP.EXC":DIM A$(
40)
20 ? "ENTER NEXT LINE (RETURN TO EXIT)
":INPUT A$
30 IF LEN(A$)=0 THEN 60
40 ? #1;A$
50 GOTO 20
60 CLOSE #1:END

0 REM LISTING 3 for Dale Lutz Tidbits
5 REM THIS PROGRAM MAKES AN AUTORUN.SY
S
6 REM FILE WHICH WILL AUTOMATICALLY
7 REM LOAD AND RUN A BASIC SAVED
8 REM PROGRAM WITH THE FILENAME "HELLO
"
10 OPEN #1,8,0,"D:AUTORUN.SYS"
20 FOR A=1 TO 155:READ I:PUT #1,I:NEXT
 A
30 CLOSE #1
50 DATA 255,255,0,6,142,6,162,0,189,26
,3,201,69,240,5,232
52 DATA 232,232,208,244,232,142,105,6,
189,26,3,133,205,169,107,157
54 DATA 26,3,232,189,26,3,133,206,169,
6,157,26,3,160,0,162
56 DATA 16,177,205,153,107,6,200,202,2
08,247,169,67,141,111,6,169
58 DATA 6,141,112,6,169,11,141,106,6,9
6,172,106,6,240,9,185
60 DATA 131,6,206,106,6,160,1,96,138,7
2,174,105,6,165,205,157
62 DATA 26,3,232,165,206,157,26,3,104,
170,169,155,160,1,96,7
64 DATA 0,251,243,51,246,67,6,163,246,
51,246,60,246,76,228,243
66 DATA 0,32,32,32,32,32,32,32,32,32,7
9,76,76,69,72,58
68 DATA 68,34,78,85,82,226,2,227,2,0,6
```

Ruth Ellsworth: JOYSTICK ROUTINES

BASIC

```
10 REM JOYSKETCH
20 GRAPHICS 7+16
30 X=80
40 Y=40
50 IF STICK(0)=7 THEN X=X+1
60 IF STICK(0)=6 THEN X=X+1:Y=Y-1
70 IF STICK(0)=14 THEN Y=Y-1
80 IF STICK(0)=5 THEN X=X+1:Y=Y+1
90 IF STICK(0)=11 THEN X=X-1
100 IF STICK(0)=10 THEN X=X-1:Y=Y-1
110 IF STICK(0)=13 THEN Y=Y+1
120 IF STICK(0)=9 THEN X=X-1:Y=Y+1
130 IF X>158 THEN X=159
140 IF Y>94 THEN Y=95
150 IF X<1 THEN X=0
160 IF Y<1 THEN Y=0
170 COLOR 1
180 PLOT X,Y
190 FOR WAIT=1 TO 5:NEXT WAIT
200 GOTO 50
```

PILOT

```
5 R:PILOT ETCH
10 C:#Y=0
20 C:#X=0
100 *STICK
110 J(%J0=1):*A
120 J(%J0=9):*B
130 J(%J0=8):*C
140 J(%J0=10):*D
150 J(%J0=6):*F
160 J(%J0=2):*E
170 J(%J0=4):*G
180 J(%J0=5):*H
191 J(%X<-79):*I
192 J(%X>79):*J
193 J(%Y<-31):*K
194 J(%Y>47):*L
240 GR:GOTO %X,%Y
250 J:*STICK
255 E:
260 *A
270 C:#X=%X
280 C:#Y=%Y+1
281 GR:DRAWTO#X,#Y
285 J:*STICK
287 E:
290 *B
300 C:#Y=%Y+1
310 C:#X=%X+1
312 GR:DRAWTO#X,#Y
315 J:*STICK
317 E:
320 *C
330 C:#Y=%Y
340 C:#X=%X+1
345 GR:DRAWTO#X,#Y
346 J:*STICK
347 E:
350 *D
360 C:#Y=%Y-1
370 C:#X=%X+1
375 GR:DRAWTO#X,#Y
376 J:*STICK
377 E:
380 *E
390 C:#Y=%Y-1
400 C:#X=%X
405 GR:DRAWTO#X,#Y
407 J:*STICK
408 E:
410 *F
420 C:#Y=%Y-1
430 C:#X=%X-1
435 GR:DRAWTO#X,#Y
437 J:*STICK
438 E:
440 *G
450 C:#Y=%Y
460 C:#X=%X-1
465 GR:DRAWTO#X,#Y
467 J:*STICK
468 E:
470 *H
480 C:#Y=%Y+1
490 C:#X=%X-1
495 GR:DRAWTO#X,#Y
497 J:*STICK
498 E:
500 *I
510 C:#X=-79
515 GR:DRAWTO-79,#Y
516 J:*STICK
517 E:
520 *J
530 C:#X=79
535 GR:DRAWTO79,#Y
536 J:*STICK
538 E:
540 *K
550 C:#Y=-31
555 GR:DRAWTO#X,-31
557 J:*STICK
558 E:
560 *L
570 C:#Y=47
575 GR:DRAWTO#X,47
577 J:*STICK
578 E:
```

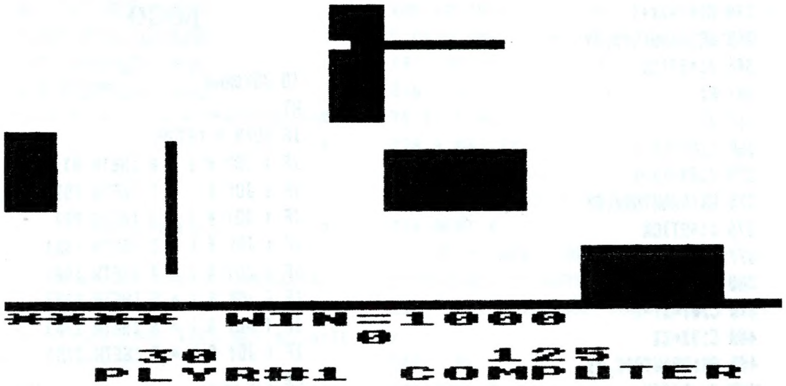
LOGO

```
 TO JOYDRAW
 HT
 IF JOYB 0 [STOP]
 IF ( JOY 0 ) = 0 [SETH 0]
 IF ( JOY 0 ) = 1 [SETH 45]
 IF ( JOY 0 ) = 2 [SETH 90]
 IF ( JOY 0 ) = 3 [SETH 135]
 IF ( JOY 0 ) = 4 [SETH 180]
 IF ( JOY 0 ) = 5 [SETH 225]
 IF ( JOY 0 ) = 6 [SETH 270]
 IF ( JOY 0 ) = 7 [SETH 315]
FD 1
JOYDRAW
END
```

ACTION!

```
PROC DRAW()

BYTE  ctr,x=[80],y=[40]

Graphics(7+16)
 DO

    IF Stick(0)=7 THEN x=x+1
    ELSEIF Stick(0)=6 THEN x=x+1 y=y-1
    ELSEIF Stick(0)=14 THEN y=y-1
    ELSEIF Stick(0)=5 THEN x=x+1 y=y+1
    ELSEIF Stick(0)=11 THEN x=x-1
    ELSEIF Stick(0)=10 THEN x=x-1 y=y-1
    ELSEIF Stick(0)=13 THEN y=y+1
    ELSEIF Stick(0)=9  THEN x=x-1 y=y+1  FI

   IF x>158 THEN x=159 FI
   IF y>94  THEN y=95  FI
   IF x<1 THEN x=0  FI
   IF y<1 THEN y=0 FI

color=1
Plot(x,y)

FOR ctr=1 TO 2000 DO
  OD

OD
RETURN
```

Stan Ockers: ANGLEWORMS (ACTION!)



```
XXXXX   WIN=1000
           0
  30             125
 PLYR#1   COMPUTER
```

```
; ANGLEWORMS II
;   Stan Ockers 12/83
; ACE Newsletter, 3662 Vine Maple
; Eugene, Or 97405 $12 year
;    FEB 1984  ACTION! Listing

MODULE  ;Some global variables
  BYTE ARRAY Xsave0 (200),Xsave1(200),
    Ysave0(200),Ysave1 (200),xpos (2),
    ypos(2),sndflg(3),pit(3),vol(3),
    dur(3),sxpos(2),sypos(2),savestk(2)
  BYTE head0,tail0,head1,tail1,stk,
    consol=53279,txtrow=656,txtcol=657,
    color1=709,color2=710,sflag0=[0],
    sflag1=[0],x,y,longcnt,grow,plyrs,
    level,shots0,shots1,win,del,tunepos
  CARD cnt,speed,score0,score1,goal
  INT ARRAY delx (2),dely (2),sdelx(2),
    sdely(2)
  BYTE ARRAY tunenote=[64 72 81 81 81
    128 121 108 108 108 81 72 64 64 64
    72 81 72 64 72 72 64 72 81 81 81
    128 121 108 108 108 81 72 64 53 53
    64 81 72 64 72 81 128 108 128 108
    128 121 108 108 121 96 121 96 121
    108 96 96 108 81 81 108 108 128
    128 144 81 72 64 53 53 64 81 72 64
    72 81]
  BYTE ARRAY tunedur=[8 8 16 16 16 8 8
    16 16 16 8 8 16 16 8 8 8 8 16 16
    16 8 8 16 16 16 8 8 16 16 16 8 8
    16 16 8 8 8 8 16 16 32 16 16 16 16
    8 8 16 32 16 16 16 16 8 8 16 16 16
    16 16 16 16 16 16 16 8 8 16 16 8 8
    8 8 16 16 16]
PROC Init() ; Worms to starting pos.
  FOR cnt=0 TO 199
  DO
    Xsave0(cnt)=10 Ysave0(cnt)=10
    Xsave1(cnt)=10 Ysave1(cnt)=10
  OD
  FOR cnt=0 TO 4
  DO
    Xsave0(cnt)=9+cnt Xsave1(cnt)=70-cnt
  OD
  FOR cnt=0 TO 3
  DO
    sndflg(cnt)=0  pit(cnt)=0 vol(cnt)=0
  OD
  xpos(0)=13 ypos(0)=10 delx(0)=1
  dely(0)=0 xpos(1)=66 ypos(1)=10
  delx(1)=-1 dely(1)=0 savestk(0)=15
  savestk(1)=15 speed=2000/level grow=3
  longcnt=grow head0=4 head1=4 tail0=0
  tail1=0
RETURN

PROC Bkgd() ; Draw walls & rectangles
  BYTE x,y,line,height,width
  CARD dl
  BYTE ARRAY dli=[72 138 72 152 72 169
    56 162 32 160 82 141 10 212 141 26
    208 142 22 208 140 23 208 104 168
    104 170 104 64]
  Graphics(5) dl=Peek(560)
  dl==+Peek(561)*256 Poke(dl+44,$8A)
  Poke(dl+45,70) Poke(dl+48,6)
  Poke(dl+49,6) Poke(dl+50,6)
  Poke(708,138) Poke(709,202)
  Poke(710,70) Poke(712,130)
  txtrow=0 txtcol=0
  FOR cnt=1 to shots0
    DO
      Put(42)
    OD
  IF plyrs=2 THEN txtcol=15
    FOR cnt=1 to shots1
    DO
      Put(170)
    OD
  FI
  txtrow=0 txtcol=6 Print("level ")
  PrintB(level)
  txtrow=1 txtcol=23 Print("plyr#1")
  PokeC(512,dli) Poke(54286,192)
  IF plyrs=2 THEN Print("  plyr#2")
  ELSE Print(" computer")
  FI
  txtrow=1 txtcol=4 PrintC(score0)
  txtcol=12 PrintC(score1)
  color=1 Position(0,0) DrawTo(79,0)
  DrawTo(79,39) DrawTo(0,39) DrawTo(0,0)
  FOR cnt=1 to Rand(3)+2
  DO
  x=Rand(80) y=Rand(40)
  height=Rand(10)+3 width=Rand(10)+3
    IF x+width>79 THEN width=79-x
    FI
    IF y+height>39 THEN height=39-y
    FI
    color=1
    FOR line=y to y+height
      DO
        Plot(x,line) DrawTo(x+width,line)
      OD
  OD
  color=0 Plot(1,10) DrawTo(78,10)
  FOR cnt=0 TO 4
  DO
    color=2
    Plot(Xsave0(cnt),Ysave0(cnt))
    color=3
    Plot(Xsave1(cnt),Ysave1(cnt))
  OD
RETURN

PROC Note (BYTE chan,pitch,duration)
  IF sndflg(chan)=0 THEN
  Poke($D200+2*chan,pitch)
  pit(chan)=pitch vol(chan)=8
  Poke($D201+2*chan,168)
  sndflg(chan)=1 dur(chan)=duration
  FI
RETURN

PROC Play(BYTE chan) ; Play tune
  IF sndflg(chan)=1 THEN
    IF dur(chan)>0 THEN dur(chan)==-1
    ELSE vol(chan)==-1
      Poke($D201+2*chan,160+vol(chan))
      IF vol(chan)=0 THEN sndflg(chan)=0
      FI
    FI
  FI
RETURN

PROC Stkupdate(BYTE stknum)
  BYTE ARRAY clank
  BYTE snd
```

```
 clank="!%*29DLUfr" snd=clank(Rand(10))
 stk=Stick(stknum)
 IF stk<>savestk(stknum) THEN
  Note(stknum,snd,1)
 FI
 savestk(stknum)=stk
 IF stk=14 THEN
  dely(stknum)=-1 delx(stknum)=0
 ELSEIF stk=13 THEN
  dely(stknum)=1 delx(stknum)=0
 ELSEIF stk=7 THEN
  dely(stknum)=0 delx(stknum)=1
 ELSEIF stk=11 THEN
  dely(stknum)=0 delx(stknum)=-1
 FI
 xpos(stknum)==+delx(stknum)
 ypos(stknum)==+dely(stknum)
RETURN

PROC Lookfwd() ;For computers worm
 IF Locate (xpos(1)+2*delx(1),
  ypos(1)+2*dely(1))<>0 OR
  Rand(100)<5 THEN
   IF delx(1)<>0 THEN delx(1)=0
    IF Rand(2)=0 THEN dely(1)=1
    ELSE dely(1)=-1
    FI
   ELSEIF dely(1)<>0 THEN dely(1)=0
    IF Rand(2)=0 THEN delx(1)=1
    ELSE delx(1)=-1
    FI
   FI
   IF Locate (xpos(1)+2*delx(1),
    ypos(1)+2*dely(1))<>0 THEN
     delx(1)=-delx(1)
     dely(1)=-dely(1)
   FI
 FI
 xpos(1)==+delx(1) ypos(1)==+dely(1)
RETURN

PROC Addhead0() ; Extend head
 head0==+1
 IF head0=200 THEN head0=0
 FI
 Xsave0(head0)=xpos(0)
 Ysave0(head0)=ypos(0)
 color=2 Plot(xpos(0),ypos(0))
RETURN

PROC Addhead1()
 head1==+1
 IF head1=200 THEN head1=0
 FI
 Xsave1(head1)=xpos(1)
 Ysave1(head1)=ypos(1)
 color=3 Plot(xpos(1),ypos(1))
RETURN
```

```
PROC Erasetail0() ; Remove from tail
 color=0 Plot(Xsave0(tail0),
 Ysave0(tail0))
 tail0==+1
 IF tail0=200 THEN
  tail0=0
 FI
RETURN

PROC Erasetail1()
 color=0 Plot(Xsave1(tail1),
 Ysave1(tail1))
 tail1==+1
 IF tail1=200 THEN
  tail1=0
 FI
RETURN

PROC Dly(CARD maxwait)
 CARD wait
 FOR wait=1 to maxwait
 DO
 OD
RETURN

PROC Title() ; Opening screen-choices
 BYTE wait=[1],choice,ypos
 CARD dl
 BYTE ARRAY firstline(14)=[13 65 238
  103 76 197 247 79 210 109 83 32
  105 105]
 DO
 UNTIL consol<>6
 OD
 Graphics(17) Poke(712,64) tunepos=0
 dl=Peek(560) dl==+Peek(561)*256
 Poke(dl+12,7) Poke(dl+8,2)
 Poke(dl+18,2)
 Position(3,8) PrintD(6,firstline)
 Position(4,10)
 PrintD(6,"by S. OCKERS")
 Position(11,14)
 PrintD(6,"Written in ACTION!")
 Position(3,23)
 PrintD(6,"fire-start game") choice=17
 DO
 wait==-1
 IF wait<=0 THEN wait=15
  Position(7,17)
  PrintBD(6,plyrs) PrintD(6," players"
  Position(7,19) PrintD(6,"level ")
  PrintBD(6,level)
  Position(7,21) PrintD(6,"goal  ")
  PrintCD(6,goal)
  FOR ypos=17 TO 21 STEP 2
   DO
    Position(5,ypos) PutD(6,32)
   OD
```

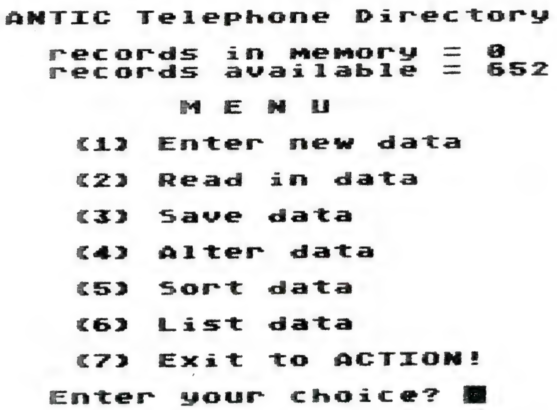
```
   Position(5,choice) PutD(6,138)
   stk=Stick(0)
   IF stk=14 AND choice>17 THEN
    choice==-2
   ELSEIF stk=13 AND choice<21 THEN
    choice==+2
   ELSEIF choice=17 AND (stk=7 OR
    stk=11) THEN
    IF plyrs=1 THEN plyrs=2
    ELSE plyrs=1
    FI
   ELSEIF stk=7 AND choice=19 AND
    level<9 THEN level==+1
   ELSEIF stk=11 AND choice=19 AND
    level>1 THEN level==-1
   ELSEIF stk=7 AND choice=21 AND
    goal<9900 THEN goal==+100
   ELSEIF stk=11 AND choice=21 AND
    goal>1000 THEN goal==-100
   FI
 FI
 Play(0) DLY(350)
 IF sndflg(0)=0 THEN
  Note(0,tunenote(tunepos),
   tunedur(tunepos))
  tunepos==+1
  IF tunepos=73 THEN tunepos=0
  FI
 FI
 UNTIL Strig(0)=0
 OD
 SndRst()
RETURN

PROC Flash0() ; Worm hit something
 FOR cnt=1 to 10
 DO
  color1=color1+8
  Sound(0,20+5*cnt,12,10) Dly(3000)
  color1=color1-8 Sound(0,0,0,0)
  Dly(3000)
 OD
RETURN

PROC Flash1()
 FOR cnt=1 to 10
 DO
  color2=color2+8
  Sound(1,50+5*cnt,12,10) Dly(3000)
  color2=color2-8 Sound(1,0,0,0)
  Dly(3000)
 OD
RETURN

PROC Sinit0() ; Set up shot movement
 sflag0=1 sxpos(0)=xpos(0)+delx(0)
 sypos(0)=ypos(0)+dely(0)
 sdelx(0)=delx(0) sdely(0)=dely(0)
```

```
    Poke(656,0) Poke(657,shots0-1)
    Put(32) shots0==-1
  RETURN

  PROC Sinit1()
    sflag1=1 sxpos(1)=xpos(1)+delx(1)
    sypos(1)=ypos(1)+dely(1)
    sdelx(1)=delx(1) sdely(1)=dely(1)
    Poke(656,0) Poke(657,shots1+14)
    Put(32) shots1==-1
  RETURN

  PROC Smove0() ; Move shot
    color=0 Plot(sxpos(0),sypos(0))
    Sound(0,60,2,10)
    FOR cnt=1 to 2
    DO
      sxpos(0)==+sdelx(0) sypos(0)==+sdely(0)
      color=0 plot(sxpos(0),sypos(0))
      IF sxpos(0)<=1 OR sxpos(0)>=78 OR
         sypos(0)<=1 OR sypos(0)>=38 THEN
         sflag0=0 Sound(0,0,0,0) EXIT
      FI
    OD
    IF sflag0=0 THEN color=0
    ELSE color=2
    FI
    Plot(sxpos(0),sypos(0))
  RETURN

  PROC Smove1()
    color=0 Plot(sxpos(1),sypos(1))
    Sound(1,70,2,10)
    FOR cnt=1 to 2
    DO
      sxpos(1)==+sdelx(1) sypos(1)==+sdely(1)
      color=0 plot(sxpos(1),sypos(1))
      IF sxpos(1)<=1 OR sxpos(1)>=78 OR
         sypos(1)<=1 OR sypos(1)>=38 THEN
         sflag1=0 Sound(1,0,0,0) EXIT
      FI
    OD
    IF sflag1=0 THEN color=0
    ELSE color=3
    FI
    Plot(sxpos(1),sypos(1))
  RETURN

  PROC Marque() ; Used in end display
    FOR x=1+del TO 16+del STEP 3
    DO
      Play(0)
      Position(x,1) PutD(6,170)
      IF x<18 THEN Position(x+1,1)
        PutD(6,42)
      ELSE Position(1,1) PutD(6,42)
      FI
    OD
    FOR y=2+del TO 20+del STEP 3
    DO
      Position(18,y) PutD(6,170)
      IF y<22 THEN Position(18,y+1)
      PutD(6,42)
      ELSE Position(18,2) PutD(6,42)
      FI
    OD
    FOR x=3-del TO 18-del STEP 3
    DO
      Position(x,23) PutD(6,170)
      IF x>1 THEN Position(x-1,23)
      PutD(6,42)
      ELSE Position(18,23) PutD(6,42)
      FI
    OD
    FOR y=4-del TO 22-del STEP 3
    DO
      Position(1,y) PutD(6,170)
      IF y>2 THEN Position(1,y-1) PutD(6,42)
      ELSE Position(1,22) PutD(6,42)
      FI
    OD
  RETURN

  PROC Gameover() ; Ending screen
    Graphics(17) Poke(712,32) tunepos=0
    Position(3,5) PrintD(6,"the winner is")
    Position(4,7)
    IF score0>=goal THEN
      PrintD(6," PLAYER #1")
    ELSEIF plyrs=2 THEN
      PrintD(6," PLAYER #2")
    ELSE PrintD(6,"THE COMPUTER")
    FI
    Position(2,9)
    PrintD(6,"WITH A SCORE OF")
    Position(4,11)
    IF score0>=goal THEN PrintCD(6,score0)
    ELSE PrintCD(6,score1)
    FI
    PrintD(6," VS ")
    IF score0>=goal THEN PrintCD(6,score1)
    ELSE PrintCD(6,score0)
    FI
    Position(4,13)
    PrintD(6,"ON LEVEL# ") PrintBD(6,level)
    Position(3,18) PrintD(6,"press START")
    Position(4,19) PrintD(6,"to play again")
    color=170 Plot(1,1) DrawTo(18,1)
    DrawTo(18,23)
    DrawTo(1,23) DrawTo(1,1) color=42
    FOR x=1 TO 16 STEP 3
    DO
      Plot(x,1) Plot(x+2,23)
    OD
    FOR y=1 TO 22 STEP 3
    DO
      Plot(1,y) Plot(18,y+1)
    OD
    del=0 cnt=1
    DO ; Flashing lights & music
      Marque() del==+1
      IF del=3 THEN del=0
      FI
      IF sndflg(0)=0 THEN
        Note(0,tunenote(tunepos),
        tunedur(tunepos)) tunepos==+1
        IF tunepos=73 THEN tunepos=0
        FI
      FI
    UNTIL Consol=6
    OD
  RETURN

  PROC Main() ; The game
    BYTE next0=[0],next1=[0]
    CARD time
    DO ; This loop is endless
    DO ; Restart game
      plyrs=1 level=1 score0=0 score1=0
      shots0=5 shots1=5 win=0 goal=1000
      DO ; Get playing choices
        Title()
        DO ; Draw a new playfield
          Init() Bkgd() time=0
          DO ; Move worms until collision
            Stkupdate(0)
            IF plyrs=1 THEN Lookfwd()
            ELSE stkupdate(1)
            FI
            Addhead0() Addhead1()
            longcnt==-1
            IF longcnt>0 THEN
              Erasetail0() Erasetail1()
            ELSE longcnt=grow
            FI
            IF sflag0=0 AND Strig(0)=0
              AND shots0>0 THEN Sinit0()
            FI
            IF sflag1=0 AND Strig(1)=0
              AND shots1>0 THEN Sinit1()
            FI
            IF sflag0=1 THEN Smove0()
            FI
            IF sflag1=1 THEN Smove1()
            FI
            Dly(speed) Play(0) Play(1)
            time=time+1 txtrow=0 txtcol=29
            IF time MOD 5=0 THEN
              PrintC(time)
            FI
            txtrow=0 txtcol=6
            IF time MOD 50=0 THEN
              Print("level ")
              PrintB(level) Print("  ")
      ELSEIF time MOD 3Ø=Ø THEN
        Print("win=") PrintC(goal)
      FI
      nextØ=Locate(xpos(Ø)+delx(Ø),
      ypos(Ø)+dely(Ø))
      next1= Locate(xpos(1)+delx(1),
      ypos(1)+dely(1))
    UNTIL nextØ>Ø OR next1>Ø
    OD
  SndRst()
  IF nextØ>Ø THEN flashØ()
  ELSE flash1()
  FI
  time=time/5*5
  DO ; Update score
    time==-5
    IF nextØ>Ø THEN score1==+5
      Note(Ø,5Ø,6)
    ELSEIF next1>Ø THEN
      scoreØ==+5 Note(1,6Ø,6)
    FI
    txtrow=1 txtcol=4 PrintC(scoreØ)
    txtcol=12 PrintC(score1) Poke(77,Ø)
    txtrow=Ø txtcol=29 PrintC(time)
    Print("     ")
    FOR cnt=1 TO 1Ø
    DO
      Dly(2ØØ) Play(Ø) Play(1)
    OD
    UNTIL time<=Ø OR time>64ØØØ
    OD
    SndRst()
    IF scoreØ>=goal OR score1>=goal
      THEN win=1 EXIT

    FI
    DO
    UNTIL Strig(Ø)=Ø OR Strig(1)=Ø
      OR consol=5 OR consol=3
    OD
  UNTIL consol=5 OR consol=3
    ; Exit only if consol key pushed
  OD
 UNTIL win=1 OR consol=3
   ; Exit on win or option key
 OD
IF consol=3 THEN EXIT ; skip gameover
FI
Gameover()
OD
OD
RETURN
```

Stan Ockers:
ATARITRAIN corrections from Dec

```
2050 SOUND 0,P,10,8:SOUND 1,2*P,10,6:F
OR L=1 TO 12*T
2060 NEXT L:SOUND 0,0,0,0:SOUND 1,0,0,
0
2070 NEXT J:RETURN
```

Clint Parker: TELEPHONE DIRECTORY (ACTION!)



```
ANTIC Telephone Directory

   records in memory  =  0
   records available  =  652

          M E N U

     (1) Enter new data
     (2) Read in data
     (3) Save data
     (4) Alter data
     (5) Sort data
     (6) List data
     (7) Exit to ACTION!

   Enter your choice? █
```

```
MODULE ; TELEPHON.ACT

; ANTIC telephone directory
; by Clinton Parker

; October 29, 1983
;
; ACE NEWSLETTER, 3662 VINE MAPLE
; EUGENE, OR 974Ø5        $12YR
;        FEB 84 ISSUE
; SEE ANTIC FEB 84 ARTICLE BY
;     JERRY WHITE FOR DETAILS


DEFINE STRING = "CHAR ARRAY"

DEFINE LNAME = "Ø",
       FNAME = "12",
       ACODE = "24",
DEFINE STRING = "CHAR ARRAY"

DEFINE LNAME = "Ø",
       FNAME = "12",
       ACODE = "24",
       EXCHG = "27",
       PHNUM = "3Ø"
DEFINE RECLEN = "34"


; records must be the first array
; declaration so that free memory
; will be calculated properly in
; proceedure Main below
CARD ARRAY records(1)

BYTE choice, IO_err
CARD max_rec, total
STRING dev, record, temp(12Ø)
```

```
; trap errors here
PROC MyError(BYTE err)
  IO_err = err
  PrintF("%EError - %B%E", err)
RETURN


PROC MyOpen(BYTE mode)
  IO_err = Ø ; assume no error
  Close(2)
  Open(2, dev, mode, Ø)
RETURN


PROC Clear()
  Graphics(Ø)
  SetColor(2, choice, Ø)
  SetColor(4, choice, 4)
  SetColor(1, Ø, 13)
RETURN


PROC Ding()
  BYTE volume
  CARD delay

  volume = 15
  DO
    Sound(Ø,choice,2,volume)
    volume = volume - 1

    FOR delay = 1 TO 3ØØ DO OD
  UNTIL volume=Ø OD
  SndRst()
RETURN


; beep user & wait until START pushed
PROC GetSTART()
  BYTE CONSOL=53279
```

```
   Ding()
   CONSOL = 0
   DO UNTIL CONSOL=6 OD
 RETURN


 PROC StartIO()
   PrintF("%Epress START when I/O
  device is ready%E")
   GetSTART()
 RETURN


 PROC StartMenu()
   PrintF("%Epress START for Menu%E")
   GetSTART()
 RETURN


 CHAR FUNC GetCh(STRING msg, CHAR c1, c2)
   CHAR ch

   DO
     PrintF(msg)
     ch = GetD(7) Put(ch)
     ch = ch % $20 ; make lowercase
   UNTIL ch=c1 OR ch=c2 OD
 RETURN(ch)


 CARD FUNC Search(CARD rec)
   BYTE inlen

   IF rec=1 THEN
     DO
       PrintF("%EEnter last name for
     search? ")
       InputS(temp)
     UNTIL temp(0)>0 OD
   FI

   IF temp(0)>FNAME THEN ; don't go
     temp(0) = FNAME     ; into first
   FI                    ; name

   FOR rec = rec TO total DO
     record = records(rec)
     record(0) = temp(0)
     IF SCompare(temp, record)=0 THEN
       record(0) = RECLEN
       EXIT
     FI
     record(0) = RECLEN ; reset length
   OD
 RETURN(rec)


 PROC GetF(STRING msg, BYTE offset, size)
   BYTE COL=85, old_col
```

```
   PrintF(msg)

 ; print out old field for default
 ; +1 is for size byte of string
   MoveBlock(temp+1, record+offset+1,
   size)
   temp(0) = size
   old_col = COL
   Print(temp)
   COL = old_col - 1
   Put(31) ; move right to show cursor

   InputS(temp)
   IF temp(0)<size THEN ; pad with blanks
     SetBlock(temp+temp(0)+1, size-temp(0),
  ,' )
   FI
 ; +1 is for size byte of string
   MoveBlock(record+offset+1, temp+1, size)
 RETURN


 PROC Modify() ; record
   Ge
 tF("%EEnter last name: ",LNAME,FNAME-
  LNAME)
   GetF("%EEnter first name: ",FNAME,ACODE
  -FNAME)
   GetF("%Earea code: ",ACODE,EXCHG-ACODE)
   GetF("%Eexchange: ",EXCHG,PHNUM-EXCHG)
   GetF("%Enumber: ",PHNUM,RECLEN-PHNUM)
   record(0) = RECLEN ; set string size
 RETURN


 PROC Enter()
   IF total>=max_rec THEN
     PrintF("%ENo additional records
      available%E")
     StartMenu()
   ELSE
     PrintF("%Eenter new data:%E")
     total = total + 1
     record = records(total)
     Modify()
   FI
 RETURN


 PROC Read()
   PrintF("%ERead data%E")
   StartIO()

   MyOpen(4)
   IF IO_err=0 THEN
     total = 0
     DO
       IF total>=max_rec THEN
```

```
         PrintE("File too big")
         Ding()
         EXIT
       FI

       InputSD(2, records(total+1))
       IF EOF(2) THEN EXIT FI

       total = total + 1
       PrintF("%Erecord #%U%E", total)
       PrintE(records(total))
     OD
   FI
   Close(2)
   StartMenu()
 RETURN


 PROC Save()
   CARD me

   IF total=0 THEN
     PrintE("I have no data to save")
   ELSE
     PrintF("%ESave data%E")
     StartIO()

     MyOpen(8)
     IF IO_err=0 THEN
       FOR me = 1 TO total DO
         PrintE(records(me))
         PrintDE(2, records(me))
       OD
     FI
     Close(2)
   FI
   StartMenu()
 RETURN


 PROC Alter()
   CHAR ch
   CARD rec

   PrintF("%EAlter data%E")

   IF total=0 THEN
     PrintE("I have no data to alter")
     RETURN
   FI

   rec = 0
   DO
     rec = Search(rec+1)
     IF rec>total THEN
       PrintF("%E%E not found%E")
       StartMenu()
       RETURN
     FI
```

```
    PrintF("%E%E%S%E", record)
    ch = GetCh("%Eis this the record to
    be altered (Y/N)%E? ", 'y, 'n)
  UNTIL ch='y OD

  Modify()
RETURN


PROC Sort()
  CARD me, v
  INT L
  STRING strpos, lv


  PrintF("%ESorting")
  IF total<2 THEN RETURN FI

  v = total
  DO
    Put('.)
    v = v / 3 + 1
    FOR me = 1 TO total-v DO
;     Put('.)
      L = me
      DO
        strpos = records(L)
        lv = records(L+v)
        IF SCompare(strpos, lv)<=0 THEN
  EXIT FI

      ; exchange records
        records(L) = lv
        records(L+v) = strpos

        L = L - v
      UNTIL L<1 OD
    OD
  UNTIL v<=1 OD
  Ding()
RETURN


PROC List()
  CHAR paper
  CARD me

  IF total=0 THEN
    PrintF("%EI have no data to list%E")
  ELSE
    PrintF("%Elist data:%E")
    paper = 0
    IF GetCh("%Etype S for screen display
       only%Eor P for for screen and printer
       's, 'p)='p THEN
      IO_err = 0
      Close(4)  Open(4, "P:", 8)
      IF IO_err=0 THEN paper = 1 FI
    FI
```

```
    Clear()
    PutE()
    temp(0) = 37
    temp(25) = '(
    temp(29) = ')
    temp(33) = '-
    FOR me = 1 TO total DO
      record = records(me)
      MoveBlock(temp+1, record+1, 24)
      MoveBlock(temp+26, record+25, 3)
      MoveBlock(temp+30, record+28, 3)
      MoveBlock(temp+34, record+31, 4)
      PrintE(temp)
      IF paper THEN PrintDE(4, temp) FI
    OD
    Close(4)
  FI
  StartMenu()
RETURN


PROC Main()
  CHAR in, CH=764
  BYTE CRSINH=752
  BYTE POKMSK=16, IRQEN=53774
  BYTE LMARGIN=82, RMARGIN=83
  CARD MEMTOP=741
  CARD i, free, old_Error
  BYTE POINTER next
  STRING tab(0)="


 ; disable BREAK key
   POKMSK = 64  IRQEN = 112

 ; setup to trap errors
   old_Error = Error
   Error = MyError

 ; get record space
   total = 0 ; no records yet
   free = MEMTOP - records
   IF free>32000 THEN ; INT divide
     max_rec = 32000/(RECLEN+3) +
       (free-32000)/(RECLEN+3) - 1
   ELSE
     max_rec = free/(RECLEN+3) - 1
   FI
   SetBlock(records, free, ' )

 ; setup string array
   next = records + max_rec+max_rec+2
   FOR i = 1 TO max_rec DO
     records(i) = next
     next = next + RECLEN + 1
   OD

 ; setup screen
   choice = 0
```

```
  Clear()
  LMARGIN = 1   RMARGIN = 39

; get device
  Close(7)  Open(7,"K:")
  in = GetCh("%Etype C for Cassette or D
  for Disk: ", 'c, 'd)
  IF in='c
    THEN dev = "C:"
    ELSE dev = "D:TELEPHON.DAT"
  FI

; menu code
  DO
    Graphics(0)
    SetColor(2,12,0)
    SetColor(4,12,4)

    CRSINH = 1 ; no cursor
    tab^ = 7
    PrintF("%E%SANTIC Telephone Directory
    %E", tab)
    tab^ = 9
    PrintF("%E%Srecords in memory = %I%E"
    , tab, total)
    PrintF("%Srecords available = %I%E",
    tab, max_rec-total)
    tab^ = 15
    PrintF("%E%SM E N U%E", tab)
    tab^ = 10
    PrintF("%E%S(1) Enter new data%E", tab)
    PrintF("%E%S(2) Read in data%E", tab)
    PrintF("%E%S(3) Save data%E", tab)
    PrintF("%E%S(4) Alter data%E", tab)
    PrintF("%E%S(5) Sort data%E", tab)
    PrintF("%E%S(6) List data%E", tab)
    PrintF("%E%S(7) Exit to ACTION!%E",
    tab)

    in = 0
    WHILE in<'1 OR in>'7 DO
      Position(10, 22)
      Put(156)
      choice = 0
      CRSINH = 0 ; show cursor
      CH = $FF ; ignore old char, if any
      Position(10, 22)
      Ding()
      Print("Enter your choice? ")
      in = GetD(7)   Put(in)
    OD

    choice = in - '0
    Clear()

    IF     in='1 THEN Enter()
    ELSEIF in='2 THEN Read()
    ELSEIF in='3 THEN Save()
    ELSEIF in='4 THEN Alter()
    ELSEIF in='5 THEN Sort()
    ELSEIF in='6 THEN List()
    ELSEIF in='7 THEN EXIT
    FI
  OD

  Graphics(0)

; restore system error routine
  Error = old_Error

; enable BREAK key
  POKMSK = 192   IRQEN = 240
RETURN
```

FEB MEETING WEDS
FEB 8, 7:30 PM

SOUTH EUGENE CAFETERIA

see the video tape by Atari of
ACE & other User Groups

PARTIAL CLEARING

by Kirt E. Stockwell

Any of you who have used DATA PERFECT or similar utility programs may have noticed an interesting programming trick. At certain points in the programs, MOST of the screen will be cleared at machine language speeds, while the top few lines remain untouched. This can be a handy technique to have on hand, as the applications for this are limited only by your imagination.

After meditating on this on and off for a week or so, I figured out what they were doing. (Figuring out how other programmers solve problems is a hobby of mine). Once I figured out how to do this, it was easy to produce a BASIC program to duplicate the effect. The slick part of this is the BASIC suboutine works as fast as the machine language routine.

The reason this routine moves so fast is the command used to clear the screen used by BASIC actually accesses the machine language subroutine in the operating system. The command PRINT (up left arrow) is the desired command. To get the (up left arrow) symbol on the screen, type [ESCAPE] [CONTROL-CLEAR]. Put this symbol in quotes following a print command, or place it in a string. When printed, this character vectors the operating system to a routine placing ZERO's in screen memory, starting at the FIRST byte of screen memory.

Now here is the fun part. The starting address of screen memory is held in two bytes located at addresses 88 and 89. To arrive at the starting address of screen memory, multiply the contents of location 89 by 256. Then add the contents of location 88. The resulting number (usually 40000 on a 48K machine) is the first byte of screen RAM. To verify this, use the number as a POKE address and place some odd character there. For instance, if your machine's starting address is 40000, you should POKE 40000,10. If a character appears in the upper left corner then you have done it right.

This should give you a fun time, just playing with the different ways of using FOR-NEXT loops to POKE things into screen memory and finding the screen changing. Now, on to the program. This is designed to be used as a subroutine. The only variable you are required to feed it is the number of lines you want left blank at the top of the screen. This information should be placed in the variable LINES. I have arbitrarily placed the program line numbers starting at 10000. This is not crucial to the operation of the program, and you are welcome to change them to suit yourself. I believe the listing of the program is fairly self-explanatory, so I will offer one final comment: What we are doing is changing the starting address of screen RAM, but only as far as the operating system knows. We aren't telling the ANTIC about the change of address because then the new address will be the actual address. We don't want to change the actual address, we only want to fool the operating system into starting the clearing routine at a place of OUR choosing. After the clearing has taken place, we return the address to its original value, and the ANTIC chip is none the wiser. Have fun.

Need a modem? In the last few days, I have received several phone calls asking me for advice for the "Best" modem for the Atari. One was from the author of an upcoming book on Networking. I have used one of the original **MicroBits** (225 3rd Ave S.W., Albany, Or 97321-2242) for quite a while and always recommend it for people who do not have an 850 interface. Last week, mine went out so I sent it in for repair. It turned out it was only a loose connection, but the owners of Microbits have always been strong supporters of ACE, and they decided to send me their brand new model — the new MPP 1000C ($150). What an improvement, although the old one was great. This connects directly into the phone line and into your joystick port, has an excellent program on cartridge included (worth $50 compared to similiar programs), works with the BIT-3 and the Austin-Franklin 80 column boards, and requires no interface as it plugs into joystick port 2. The biggest improvements are **Auto-Dial and Auto-Answer** capabilities, a feature unusual in any modem of this price range. It also will store 10 phone numbers to call, download any size file, with direct disk and printer data transfer. The "Smart Terminal" cartridge allows you to prepare text and messages with its built in editor off line; there are also 9 dynamic buffers to use to up — or down-load mutilple files. Many more new features — look no more for a modem!

ACTION!

In this issue are a number of articles written in the new ACTION! language. It was not planned that way; it is because all of our regular contributors independently tried the language, liked it so much they started programming in it, and wish to share their excitment with the rest of ACE. Stan's program is a nice, fast-paced game illustrating many of the features of the language; Ruth shows how ACTION! and other languages compare; and the originator and developer (inventor?) of ACTION!, Clint Parker, has sent us many programs to put in this and future issues. The first is a companion program to the article and BASIC program of Jerry White in the FEB or Mar issue of **ANTIC!**. The original article was to be a comparison of a BASIC Telephone directory and a similiar one in ACTION!, but there wasn't room for the ACTION! listing, so Clint sent it to us. It runs about 200+ times faster that the BASIC version, so I think we got the better part of the deal. Please read the ANTIC! article for the explanation on how it works, etc.

ACTION! is manufactured by O.S.S. (1173-D Saratoga-Sunnyvale Rd., San Jose, CA 95129, $99) and comes on a bank-switched 24K cartridge. It is a joy to use, with its built in Editor and compiler. According to the company, most of the older ACTION! cartridges have bugs in it, and you can exchange yours for a new one, with up-dated documentation for $20 — now still in the EPROM version, but in about 6 weeks in a new, updated ROM version. Also available now is the **ACTION! Programmers Diskette**, full of useful utilities, etc for $30, and a new ACTION! Newsletter if you send in your Software License Agreement. If you want to get an ACTION! cartridge, expect a slight wait — they are expensive to make, and they are selling them faster than they can make them. There is also a Run-time package allowing ACTION! programs to run on a binary load file without the Cartridge for $30 (non-commercial) License fee.

-- M. Dunn

```
10000 REM **** PARTIAL CLEARING ****
10010 REM *    FOR EUGENE A.C.E.    *
10020 REM *          BY             *
10030 REM *   KIRT E. STOCKWELL     *
10040 REM *        PRESIDENT        *
10050 REM **************************
10100 LSB=PEEK(88):MSB=PEEK(89)
10110 SCR=LSB+(MSB*256)
10120 OFSET=LINES*40
10130 NSCR=SCR+OFSET
10140 NMSB=INT(NSCR/256)
10150 NLSB=NSCR-(NMSB*256)
10160 POKE 88,NLSB:POKE 89,NMSB
10170 ? "}"
10180 POKE 88,LSB:POKE 89,MSB
10190 POSITION 0,LINES
10200 RETURN
```

REVIEWS

Microbits Peripheral Products
by Kirt E. Stockwell

MPP 1000

Telecommunications has become increasingly popular lately. Our own bulletin board system (BBS) has been receiving calls at a rate exceeding 100 ... The Microbits MPP1000c modem runs WITHOUT any interface, and without any modifications to the computer. Everything required to begin telecommunication is included in the package. The package includes the MPP1000c auto-dial/auto-answer modem, Smart Terminal program cartridge, power supply with 5-foot cord, instruction manual, usual warranty paperwork, free trial membership on THE SOURCE, modular telephone cord (aprox 8 feet).

The MODEM plugs into the #2 joystick port at the computer end. On the other end, it plugs into the wall plug (or an adaptor or splitter) without any other equipment needed. Of course you can use a phone with it, but it can be ANY phone of reasonably current technology.

The Smart Terminal program (on cartridge) has the following capabilities: Full scale Uploading and Downloding; 10 dynamically allocated buffers; ASCII/ATASCII, STop-bit, Parity, and Checksum checking; Screen Editor Buffer Loading & Editing; Disk Directory; Variable Baud rate (110 to about 600); AUTO-DIAL and AUTO-ANSWER functions; and the Christiansen x-MODEM protocol.

The Auto-Dial function allows the input and saving of up to 10 phone numbers on disk which are automatically loaded when the program boots up.

The Auto-Answer function allows the MODEM to answer the phone. This DOES NOT allow you to use the MODEM as a BBS. The program is set up to automatically transmit ONE FILE each and every time the phone rings. I plan to use this feature to get my articles to the newsletter editor when I'm pushing deadline and can't be available.

Another new feature I like is the Christiansen X-Modem protocol. Generally, data is transmitted one byte at a time. Depending on whether the computers involved are set to half duplex or full duplex, the receiving computer may or may not echo the data it received. (ECHO: send back to the source) In any case, very little checking is normally done to insure one computer actually got what the other computer sent. The X-MODEM protocol causes the sending computer to transmit 128 bytes at a time. After transmission, the sending computer waits for the receiving computer to echo the ENTIRE 128 byte block. If the Echo Block is IDENTICAL to the sent block, the next block will be sent. If, on the other hand, there is even one BIT different, the process begins again, sending each block as many times as necessary to complete a flawless transfer of the entire file. While this takes a little longer it is definitely worth the extra phone charges when you have sensitive data.

The Baud rate is variable up to about 600 but most phone lines around the country can not maintain integrity transferring data above about 450 baud. The higher speeds are all right for text, but not programs or sensitive data files. The MODEM doesn't have a port for a printer or any such googaws. It is simply an extremely easy to use communications device supplied with the best smart terminal program available. The documentation is clear and thorough, and the program is VERY easy to use as well as being very powerful and flexible. I rate this a BEST BUY in the telecommunication field.

MicroFiler

MicroFiler dbms by Microbits Peripheral Products of Albany, Oregon is written by Jim Harrison, in machine language and is on a cartridge. This will be of benefit to users with minimal systems as well as those who are not sophisticated in the use of disk drives. One of the important features of this program is the storage and retrieval of data (files) can be done with cassette OR disk. Thus, those of you who start on a cassette system can graduate to disk without having to reenter all the files.

Another nice feature Jim has managed to modify is the cassette storage parameters. The cassette speed has been increased to twice the normal ATARI speed. Also, the program is designed to encourage saving files twice.

MicroFiler does not have the complex mathematical functions of programs such as Data-Perfect (my personal favorite) but it does have very good search and sort capacity. This includes multiple field searches and wild card search. Another feature I like is the ability to place comments and/or headings directly on the data screen itsself (itself?).

The program keeps track of the number of records you are permitted to enter (this of course depends on your machine's RAM and the size of the records) and tells you how many empty spots you have left. The records are kept in RAM, so there are definite limits to the number of records you have space for. On the other hand, all sorting and searching is handled quickly since nothing needs to be fetched from disk or tape.

MicroFiler is simple to use and very well documented. I recommend this one to most of you who haven't yet taken to using database managers. You might want to try this one. It is a basic yet feature-packed program giving you the power you need for your beginning level applications, as well as being a convenient program for keeping simple records on. This is an excellent program with which to break into the world of database management. The price is also very reasonable, especially when you take into account that the program is on cartridge.

A.B.I.S.

(Atari BASIC Interactive Structuring)

ABIS is a program for writing Atari BASIC programs which display textual material. It can be used by the programmer and nonprogrammer alike. It allows one to easily format text on the screen using the familiar Atari screen editor, then it writes out a BASIC program which will present the text.

The program produced features automatic paging, and may be easily modified if necessary. Its primary use is for making tutorials to be run on a computer. I have used ABIS to make programs to present the documentation for other programs, usually on the same disk. In this way, instructions can be provided without the need for paper. A very good application of this program, therefore, is to document programs on public domain diskettes so potential users will have an idea how to use programs whose function is less obvious. I have also used ABIS to make menu and instruction screens for use in other BASIC programs. When this is done, however, some modifications to the produced program are necessary.

The Program

When typing in the program, you will have to be quite careful about the positioning of quotation marks. Since ABIS is a program for writing other programs, the lines may seem contorted, but rest assured that if you enter them in correctly, it will work. The main loop is from lines 20 to 30 (actually ridiculously short). The reason this is so is that we are using the Atari screen editor which has so many functions built in. Lines 100 to 140 do the initialization, and lines 145 to 190 write out the first routines of the tutorial program. Variables LINE and INC in line 100 control at what line number the program will start and by how much each line will go up by, respectively. You may change these if you wish. Lines 200 to 280 write out the routine to produce the text on the screen, and lines 300 to 330 are executed when the program execution is terminated. Note that in line 185 the break key is disabled, so it may not accidently be pressed, causing a disaster.

Using ABIS

Before using ABIS, you must first specify where the program produced is to be saved. If you press C for cassette, it will be written out to tape. You will need to correctly position the tape, press PLAY and RECORD, and then press RETURN.

If disk is chosen, you will need to enter a filename. Note that the usual "D:"prefix is not required.

Now you are ready to format the first page of text. Use the normal screen editor functions to place text on the screen. (Note: All features of the screen editor are functional, with the notable exception of the CLEAR key. This has been disabled to prevent the accidental loss of a whole screen of text. If you wish to clear the screen at any point, simply position the cursor on the top line using the arrow keys, and then press SHIFT-DELETE as necessary.

Some formatting tips: To center a line of text, type it near the beginning of the line, then use CONTROL-INSERT and CONTROL-DELETE to get it to where it looks right. Also, press RETURN when you're fairly close to the right margin so you don't need to do lengthy CONTROL-INSERTs to effect word wraparound. I like to double space my text (I think this makes it easier to read). Also, I always like to leave a little message on the bottom of the screen to prompt the reader to push any key to continue.

When you are satisfied with the screen, press the SELECT button, and, while holding it down, press the OPTION button. The screen will go blank while the program is written, then it will come back on to display the finished frame. Pushing any key will allow you to enter the next page.

When you have finished entering the last page of your tutorial, hold down the START and SELECT buttons, and, while doing so, press the OPTION button. After this frame is written out and you have pressed any key, you will be asked to confirm that the last page of the tutorial has been finished. Answering anything but yes here will allow you to add more material.

If you indicate that you are indeed finished, ABIS will erase itself and ENTER in the program just written. (Cassette users will need to rewind the tape at this point.) Therefore, the program just produced will be in memory and may be SAVEd, LISTed, RUN, or edited.

All tutorials produced with ABIS feature a built in paging capability. As the reader finishes each page, he/she may press almost any key to advance to the next page. Pressing the '+', back arrow, or Delete/Back Space key will get the reader to the previous page. This allows one to review material halfway through the session.

Quirks of ABIS

First of all, ABIS disregards whatever you type on the bottom line of the screen, so don't put anything there. Also, if you go to the bottom line and continue pressing RETURN, the top of your text will scroll off and be lost forever. ABIS will allow you to place any characters on the screen except the quotation mark (Shift-2), so you can't use it (use Shift-7 instead).

A final note— ABIS will display a flashing block wherever you leave the cursor when you finish a frame, so keep that in mind. This is to prompt the user that the program is waiting for him/her to take action.

Since ABIS produced programs are linear in design, it is a good idea to make an outline of the points you wish to cover in your tutorial before sitting down to use ABIS. This usually results in a far more coherent tutorial— otherwise it is easy to forget what you have and have not covered.

Conclusion

Let me know what you think of this program. If you feel automatic centering/word wraparound is a needed feature, or you come up with routines to do this, let me know.

— Dale Lutz
Holden, CANADA

ACE EXCHANGE LIBRARY

The past couple of months have been extremely busy for the exchange library. Three new disks were added at Christmas bringing our total to 31 disks containing over 325 programs

The ERACE group has added Education disk #5. The editors have added the Best of ACE, Vol. 7 containing games from our members in England, Australia, and Stan Ockers. Ruth Ellsworth has completed Pilot #2 filled with games and projects for the younger computerists.

The complete library list has been revised as of 1-15-84 and you may obtain your own copy by sending 50 cents in stamps or coin (US $1.00 overseas) to ACE Exchange Library, 374 Blackfoot Street, Eugene, OR 97404.

The December issue of **ANTIC** magazine contained a very nice review of our disk Best of ACE 1983-#1 (now known as Best of ACE, Vol. #5). Thanks to this article, we were swamped and for awhile were running about 2 weeks late in getting orders back in the mail. We are finally getting caught up and apologize for the delay some of you experienced. For those of you using cassettes, we will still be running about 2 weeks behind on your orders. We are having the tapes professionally duplicated and we try to check each one to be sure it will load before we send it out.

One of the things mentioned in the article in ANTIC was the lack of documentation for public domain software. This is a problem we've been working on and we now have documentation available for Educational disks 1 thru 5, Pilot 1 & 2, Best of ACE, vols 4 thru 7, Utility #1, Games 1 & 2, and SCOPY. More disks (and tapes) will be documented as time permits.

During the past several months, a few people have sent copyrighted programs they wanted to exchange for material in our library. We want to ask those of of who are writing your own programs to put REM statements at the beginning of the program listing which include your name, address, and the words "In the Public Domain". If you wish to copyright the program, and yet share it with our members, then state that fact in the REMs. We will not exchange copyrighted material without license to do so. Please play fair! If YOU don't hold the copyright, don't send the program to us. It just costs us both time and money to send it back and forth.

— Aaron C. Ness
— Ronald J. Ness
Librarians

ANGLEWORMS II

When I got my Atari, one of the first games I wrote was called "Angleworms". It's the classic 'Blocade' type game where you try to keep from running into the other guy's worm, your own or walls etc. It was written in Basic and was slow but enjoyable, especially if the two players were of about equal ability. I often wanted to speed up the game but dreaded the thought of all the assembly language and debugging necessary. Then I recently purchased an Action! cartridge by OSS.

Action! is a natural language for arcade style games. The language is compiled and fast! Instead of worrying about whether things run fast enough, I now have to put in delay loops between plotting of each pixel. The highly structured form takes a little getting use to. You have to put a lot more thought into writing the program but the result is actually readable. You can return to the program a month later and follow it without spending an hour to outline the structure.

You can get the flavor of Action! by looking over the source code of Angleworms II. The primary building block is a subroutine-like structure called the procedure or PROC. Any parameters passed to the procedure follow the name in parenthesis. The procedure called when the program is run is the last one, the one I've labeled 'Main()'. Other PROCs are called by simply naming them (they must have been defined previously).

The structure turns out to be loops within loops. The indentation helps you follow what constitutes a loop. Loops are also marked at the beginning and end by DO and OD (DO backwards). If the loop is to be repeated a definite number of times, the DO is preceded by a FOR ... TO ... STEP statement. You can also get out of the loop with an EXIT statement, usually conditional, (part of an IF ... THEN EXIT). Other loop control statements are the WHILE which appears just before the DO or the UNTIL appearing just before the OD. These cause the loop to repeat WHILE a condition is true or UNTIL a condition becomes true.

Another major construct is the statement beginning with IF and ending with FI (IF backward). Following the IF is an expression, a THEN and maybe an ELSE or even ELSEIF ... THEN. If the expression evaluates as true, all statements following the THEN up until the ELSE are executed. If not, all statements following the ELSE are used. The ELSEIF ... THEN allows more conditional expressions without going to another IF..FI construction. IF's will often be nested and the indentation helps you follow which condition is active at the time.

All variables must be declared and there are three fundamental types, BYTE, CARD and INT. Sorry, no floating point so forget about those trig functions etc., just integers 0-255 BYTE), 0-65,535 (CARD) and -32768 to 32767 (INT). You have to be careful, especially when subtracting numbers from BYTE or CARD values; they 'roll over' instead of going to negative numbers. IF variable
0 THEN ... has no meaning if variable is CARD because CARD is never negative. Another problem is fractions ... there are none! If you have an expression and part way through it evaluates to a fraction, you will lose it because the fraction will be treated as zero. The operator MOD (which gives the remainder of integer division) helps somewhat.

If variables are to be valid throughout all PROC's they should be defined globally under the heading MODULE, otherwise they can be defined at the beginning of the PROC to which they apply. When defined they can also be initalized by putting an equal sign and the value in brackets. Besides the fundamental types there are the extended types of POINTERS, ARRAYS and RECORDS. I found byte arrays to be most useful. I use them as I used strings in Basic. At the beginning of the program you can see the tune notes and duration of notes defined as byte arrays.

I've not included many comments in the source because it was already getting so long. Programs tend to give long listings because of the indentation and spaces necessary to outline the structure. You could do away with a lot of this but it tends to get unreadable. The editor has a find command which helps you get to any portion of the source quickly. The editor also allows very long line lengths (240 characters) and lets you shift the viewing window over past 40 columns as well as shifting individual lines left when you enter more than 40 columns. I find it difficult to keep track in my mind of what I can't see on the screen and end up with very jagged listings, wasting even more space when printed. So, I've arbitrarily limited myself to 40 columns.

A really great feature of Action! is the fact that it is on cartridge, flipping back and forth between the editor and compiler is instantaneous — no disk accesses. Both the source and compiled object code are resident in memory at the same time and I've found it difficult (but not impossible) to clobber the source. I suppose a program which doesn't use any of the Action! library might run as a binary file. You are not likely to write a program without using at least one library routine which include most input, output, graphcs and sound statements like Basic plus some additional 'goodies' like SndRst, Zero, Setblock and Moveblock. This means you need the Action! cartridge to run the programs.

The editor is a screen editor and better than the Basic editor. You can even have a split screen, working in two different windows. I find the cut-and-paste feature especially useful. The compiler is fast, taking only a few seconds to compile most programs. The source of 'Angleworms II' takes 99 disk sectors while the object takes 68. One reason for the length of the program are many embelishments I made to the original. I had the rudimentary program working when the source was only 25-30 sectors long. Because of the cartridge, Action! is one of the few higher level languages I might consider using with a cassette based system.

Angleworms can be played by two players or by one against the computer. Using joystick 1 you can choose the number of players, difficulty or final score. Move vertically to select the variable and left or right to change the value. Press fire to start the game. The second player uses joystick #2. You can't run into anything including your own worm (you can't 'go back' on yourself). The worms will gradually lengthen as time goes on. You have five shots (fire button) which will blast a path through anything, (use them sparingly). After each collision the score of the player who didn't collide is increased depending on the time elapsed. Pressing SELECT at this time will return you to the choices so you can change the difficulty level or final score in midgame. Pressing OPTION will start a new game.

I've still neglected to mention a lot of the features of Action! I especially like the ability for bit manipulation including left and right shifts as well as the ability to assign memory locations directly to variables. Notice the assignment of consol to location 53279 in the global variables. After this assignment the consol keys can be used directly in a conditional statement like 'UNTIL consol = 5'. Machine language can be inserted also. I've used a display list interrupt sequence to change colors for the bottom part of the screen (see PROC Bkgd()). By the way, I have not implemented the attract mode for this process so don't leave the program unattended for a long time.

In summary, I'm quite pleased with Action! It's perfect for designing arcade style games and the library of routines makes programming them quite straight forward. The structured style takes some getting use to but makes the code more readable. The best thing though is the fast speed! Being on cartridge, everything is immediately available, a real blessing when working with a compiled language.

— Stan Ockers

ERACE

ACE Educational disk N0.5

Our educational disk #5 is now ready. There are some very good programs on it for students of all ages. There are spelling and counting games for the younger set, and there are word and math drills to challenge just about anyone. We want to thank those who have taken the time to write these programs and send them to us, for without them these disks could not be possible.

Following is a list of the programs on the disk.

OLDMAC.MAY — By Stan Ockers — A counting and matching game for pre-schoolers and early primary grades.

ATRAIN — By Stan Ockers — An animated spelling game for lower primary grades (K-2).

ALGEBRA1 — By John R. Kelley — An elementry algebra drill to test knowledge of the distributive axiom of multiplication with respect to addition.

ALGEBRA2 — By John R. Kelley — Another algebra drill. Determine the bi-nomial roots of simple polynomials.

SCRAMBLE.JK — By John R. Kelley — Unscramble the letters to spell the word correctly.(6th grade-adult)

WORDTROUB.LE — By Wayne Real — A homonym word game requiring you to put the correct word in a sentence.

EXPANSION — By Wayne Real — Determine the value of each "house" of a number, then add them up to arrive at the original number.

COUNTING — By Wayne Real — A fun addition or counting game for 1st or 2nd graders. Answer the problem correctly and be visited by a friendly alien.

ESTIMATI.ON — By Wayne Real — Visually assess the size, number, and co-ordinate relationships between lines, shapes,and dots. 4th grade-adult

We have recieved some programs from a sister club in Australia to help us start another disk, so keep those programs commimg and we'll see what happens on disk #6. Have fun learning until nest time.

— Robert Browning

ATARI PLAYER-MISSILE GRAPHICS IN BASIC

by Philip C. Seyer

For Basic programmers who have wanted to try adding some player-missile graphics to their programs, but have shied away because it seemed just too complicated, take heart, this one's for you.

I have read many articles and other books on graphics which often talked about PM graphics, but never have I read a book which is so easy to understand yet so thorough.

This book is written for the beginning to intermediate programmer who wants to know how to begin using the PM graphics capabilities of the Atari. It assumes you know some of the basics of programming, yet you are guided through each necessary step of PM programming by well explained text, diagrams, questions, and sample programs. It shows you how to design and store a player image. The author's use of strings makes animation simple (although there are better ways to handle the strings), and the joystick routines are simple.

There are subroutines for adding sound, detecting collisions, and player set up. Most of the memory locations needed for PM programming are noted and explained.

You will not get machine speed animation using the routines from the book, but that is not the author's intention. He lets you know there are machine subroutines which can improve program speed. Once you have completed the book you will be able to read those articles on "fast basic PM graphics" and understand them.

For those of you who have wanted to try PM graphics in BASIC, but you haven't been able to put together the bits and pieces from all the articles and books you've read, try this one you'll like it.

— Robert Browning

FUNCTION KEYS

If you've had your Atari for awhile, and have begun to do more advanced programs, you have probably noticed the Atari lacks function keys. This can be particularly annoying if you need to print a command without typing it. This program can help relieve this annoyance.

Type in the Function Key Editor and SAVE it. Then RUN. the prompts should be self-explanatory. If you don't use the defaults, just enter the commands for the desired function keys. After all the prompts are done, the program saves a file under the filename you designate. Then press SELECT to reboot the computer or press START to end the program. If you saved the file under AUTORUN.SYS then reboot the Atari. If you didn't, then go to DOS and use the Binary Load option.

To use the program, first type CTRL A and press START, SELECT or OPTION. If you typed the program correctly, the computer will respond with the message for that key. I know your next question, "Why do we have to press CTRL A?" Well, the console keys use location 53279, which is also used for the OS keyclick routines. We use the CTRL A to verify you are going to press one of the console keys. If you accidentally type or print CTRL A, don't worry. Just make sure you don't press any of the console keys until you type or print something else. If we didn't use the CTRL A, the function key operation would be very erratic. One word of caution, this program uses immediate mode VBI in Page 6, so watch out for other VBI routines!

Note: This program is RESET protected so the only way to turn it off is to go to DOS and use the B option. If the screen flickers when you use this routine, don't worry, it is normal and won't hurt anything.

— Greg Menke

Best of ACE books

Volume 1 are bound issues of the ACE Newsletter from the first issue, Oct 80 to June of 1982

Volume 2 covers July 1982 to June 1983

Only $12 each ($2 extra for Airmail). Available only from:

George Suetsugu
45-602 Apuapu St
Kanoehe, HI 96744

SortFinder 1.2

Composite index of Atari related articles from 5 popular computer periodicals from Apr '81 to June '83, including ACE. Only $6 for ACE member from:

Jim Carr, Valley Soft
2660 S.W. DeArmond
Corvallis, OR 97333

Atari
Computer
Enthusiasts
3662 Vine Maple Dr. Eugene OR 97405

FIRST CLASS MAIL

Atari Computer Enthusiasts

A.C.E. is an independent, non-profit and tax exempt computer club and user's group with no connection to the Atari Company, a division of Warner Communication Company. We are a group interested in educating our members in the use of the Atari Computer and in giving the latest News, Reviews and Rumors.

All our articles, reviews and programs come from you, our members.

Our membership is world-wide; membership fees include the A.C.E. Newsletter. Dues are $12 a year for U.S., and $22 a year Overseas Airmail and include about 10 issues a year of the ACE Newsletter. **Subscription Dep't:** 3662 Vine Maple Dr., Eugene, OR 97405.
**

President ....Kirt Stockwell
4325 Sean , Eugene, OR 97402 / 503-689-5355
Vice Pres..Larry Gold
1927 McLean Blvd., Eugene, Or 97405 / 503-686-1490
Secretary ....Bruce Ebling
1501 River Loop #1, Eugene, OR 97404 / 503-688-6872
Librarian.................................Ron and Aaron Ness
374 Blackfoot, Eugene, Or 97404 (503)689-7106.
Co-Editor ....Mike Dunn
3662 Vine Maple Dr., Eugene, Or 97405 / 503-344-6193
Co-Editor ....Jim Bumpas
4405 Dillard Rd., Eugene, Or 97405 / 503-484-9925
E.R.A.C.E. (Education SIG Editor).....................Ali Erickson
295 Foxtail Dr., Eugene, Or 97405 / 503-687-1133
E.R.A.C.E. Corresponding Secretary ....Robert Browning
90 W. Myoak, Eugene, OR 97404 / (503)689-1513

Send 27c stamps or coin (50c overseas) to the Ness' for the new, updated ACE Library List —new in Feb 84!

Bulletin Board
(503) 343-4352

On line 24 hours a day, except for servicing and updating. Consists of a Tara equipped 48K Atari 400 with a TARA keyboard, 2 double-density double sided disk drives with an ATR 8000 interface, 2 double density Percom disk drives, an Epson MX80 printer, a Hayes SmartModem; running the ARMUDIC Bulletin Board software written by Frank L. Huband, 1206 N. Stafford St., Arlington, VA 22201. See the Nov '82 issue for complete details.